AD-A052 799 TRACOR INC AUSTIN TEX F/G 8/8

COMPARISON OF NUMERICAL AND PHYSICAL HYDRAULIC MODELS, MASONBOR--ETC(U)

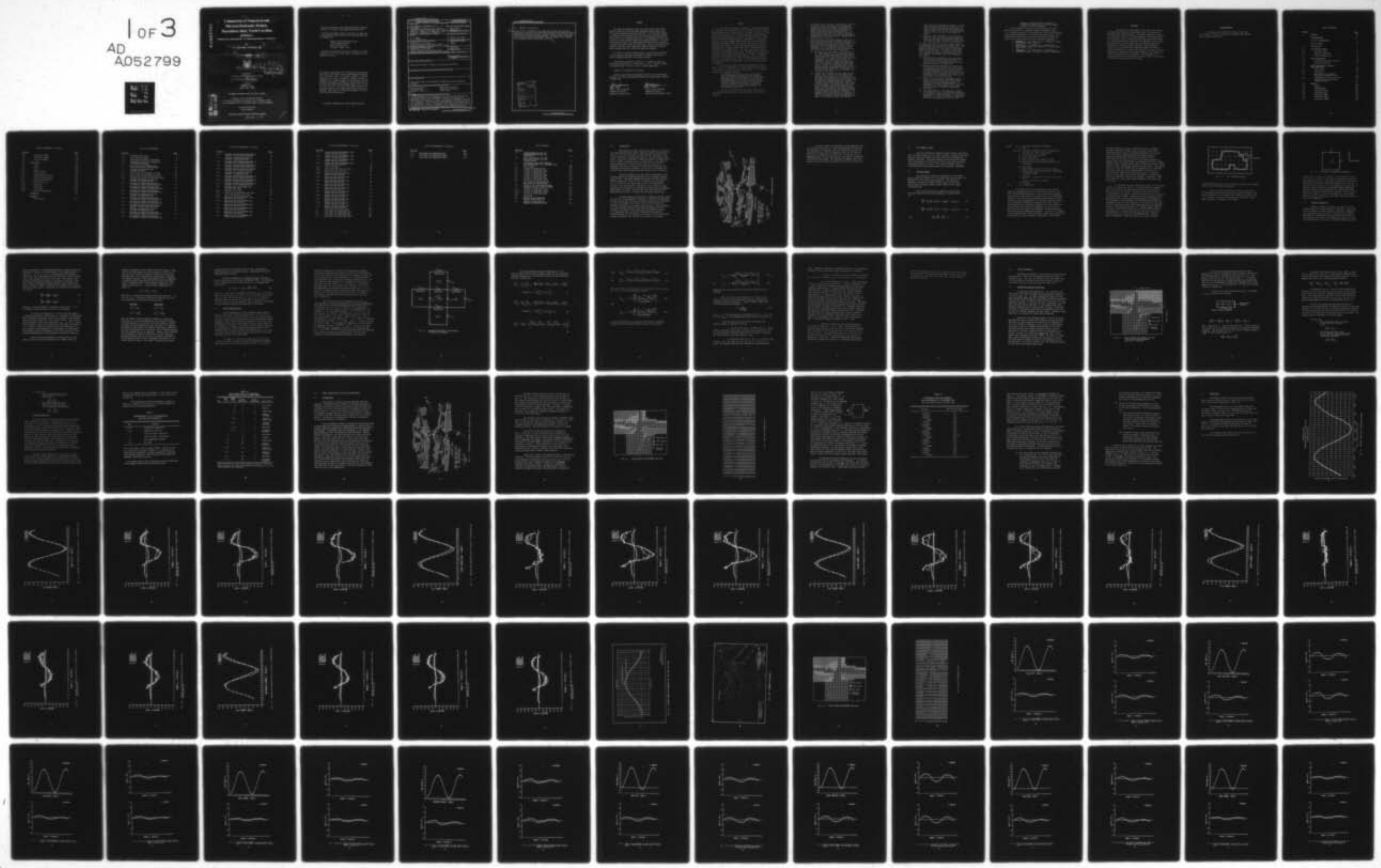
JUN 77 R J CHEN, L A HEMBREE DACW72-72-C-0029

UNCLASSIFIED CERC-GITI-6-APP-3 NL

1 OF 3

AD

A052799



6

Comparison of Numerical and Physical Hydraulic Models, Masonboro Inlet, North Carolina.

2 8

APPENDIX 3.

Numerical Simulation of Hydrodynamics (Tracor).

by

10 R. J. Chen and L. A. Hembree, Jr

9 Final rpt,

18 CERC

19 GITI ~~REPORT~~ 6 -APP-3

11 June 1977

12 281p.

Prepared for
U.S. Army Coastal Engineering Research Center
under
15 Contract DACW72-72-C-0029
by
TRACOR
6500 Tracor Lane
Austin, Texas 78721

GENERAL INVESTIGATION OF TIDAL INLETS

A Program of Research Conducted Jointly by
U.S. Army Coastal Engineering Research Center, Fort Belvoir, Virginia
U.S. Army Engineer Waterways Experiment Station, Vicksburg, Mississippi

Department of the Army
Corps of Engineers

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

352 100

mt

Limited free distribution within the United States of single copies of this publication has been made by this Center. Additional copies are available from:

National Technical Information Service
ATTN: Operations Division
5285 Port Royal Road
Springfield, Virginia 22151

Cover Photo: Masonboro Inlet, North Carolina, 24 July 1974

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER GITI Report 6, Appendix 3	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) COMPARISON OF NUMERICAL AND PHYSICAL HYDRAULIC MODELS, MASONBORO INLET, NORTH CAROLINA: APPENDIX 3, NUMERICAL SIMULATION OF HYDRODYNAMICS (TRACOR)		5. TYPE OF REPORT & PERIOD COVERED Final report Appendix 3
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) R. J. Chen L. A. Hembree, Jr.		8. CONTRACT OR GRANT NUMBER(s) DACW72-72-C-0029
9. PERFORMING ORGANIZATION NAME AND ADDRESS Department of the Army Coastal Engineering Research Center Kingman Building, Fort Belvoir, VA 22060		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS F31019
11. CONTROLLING OFFICE NAME AND ADDRESS Department of the Army Coastal Engineering Research Center Kingman Building, Fort Belvoir, VA 22060		12. REPORT DATE June 1977
		13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

This report has four appendices, published as four separate volumes.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Hydraulic models
Hydromechanics
Masonboro Inlet, N. C.
Mathematical models
Numerical simulation
Tidal flats

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The purpose of this project was to modify an existing two-dimensional numerical hydrodynamic model to be applied to Masonboro Inlet, North Carolina. The model essentially consists of a numerical solution to the time-dependent linearized shallow-water equation. One of the modifications was the incorporation of tidal flats which are important in the Masonboro Inlet system. The second was to devise a procedure for handling open water boundaries at which no boundary conditions were available.

next page

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. ABSTRACT (Continued).

cont.
Using data for September 1969 supplied by CERC, the friction factor was adjusted to simulate the observed tide stage variation. Satisfactory verification was obtained against tidal stage variation and intratidal current phasing. The magnitude of the computed currents were consistently small but this is not thought to seriously delimit the model's utility.

FOREWORD

This report was prepared by Tracor, Inc., Austin, Texas, under Contract DACW72-72-C-0029 with the US Army Coastal Engineering Research Center as part of the General Investigation of Tidal Inlets (GITI). The GITI research program is under the technical surveillance of the US Army Coastal Engineering Research Center (CERC), and is conducted by CERC, The U.S. Army Waterways Experiment Station (WES), and other government agencies and private organizations. This report contains detailed results of a numerical model developed as part of an elevation of physical and numerical models of a tidal inlet. Details of the evaluation are contained in the basic report "Comparison of Numerical and Physical Hydraulic Models, Masonboro Inlet, NC" to which this report is an Appendix.

This report is published, without change, as received from the contractor in 1973. Results and conclusions are those of the authors and are not necessarily accepted by CERC, WES, or the Corps of Engineers.

The report was prepared by R. M. Chen and L. A. Hembree of Tracor, Inc. CERC contract monitors were B. R. Bodine and D. L. Harris. Technical Directors of CERC and WES were T. Saville, Jr. and F. R. Brown, respectively.

Comments on this publication are invited.

Approved for publication in accordance with Public Law 166, 79th Congress, approved 31 July 1945, as supplemented by Public Law 172, 88th Congress, approved 7 November 1963.

JOHN L. CANNON
Colonel, Corps of Engineers
Commander and Director
Waterways Experiment Station

JOHN H. COUSINS
Colonel, Corps of Engineers
Commander and Director
Coastal Engineering Research Center

PREFACE

1. The Corps of Engineers, through its Civil Works program, has sponsored, over the past 23 years, research into the behavior and characteristics of tidal inlets. The Corps' interest in tidal inlet research stems from its responsibilities for navigation, beach erosion prevention and control, and flood control. Tasked with the creation and maintenance of navigable U. S. waterways, the Corps routinely dredges millions of cubic yards of material each year from tidal inlets that connect the ocean with bays, estuaries, and lagoons. Design and construction of navigation improvements to existing tidal inlets are an important part of the work of many Corps offices. In some cases, design and construction of new inlets are required. Development of information concerning the hydraulic characteristics of inlets is important not only for navigation and inlet stability, but also because inlets, by allowing for the ingress of storm surges and egress of flood waters, play an important role in the flushing of bays and lagoons.

2. A research program, General Investigation of Tidal Inlets (GITI), was developed to provide quantitative data for use in design of inlets and inlet improvements. It is designed to meet the following objectives:

> To determine the effects of wave action, tidal flow, and related phenomena on inlet stability and on the hydraulic, geometric, and sedimentary characteristics of tidal inlets; to develop the knowledge necessary to design effective navigation improvements, new inlets, and sand transfer systems at existing tidal inlets; to evaluate the water transfer and flushing capability of tidal inlets; and to define the processes controlling inlet stability.

3. The GITI is divided into three major study areas: (a) inlet classification, (b) inlet hydraulics, and (c) inlet dynamics.

a. The objectives of the inlet classification study are to classify inlets according to their geometry, hydraulics, and stability, and to determine the relationships that exist among the geometric and dynamic characteristics and the environmental factors that control these characteristics. The classification study keeps the general investigation closely related to real inlets and produces an important inlet data base useful in documenting the characteristics of inlets.

b. The objectives of the inlet hydraulics study are to define the tide-generated flow regime and water-level fluctuations in the vicinity of coastal inlets and to develop techniques for predicting these phenomena. The inlet hydraulics study is divided into three areas: idealized inlet model study, evaluation of state-of-the-art physical and numerical models, and prototype inlet hydraulics.

   (1) The idealized inlet model. The objectives of this model study are to determine the effect of inlet configurations and structures on discharge, head loss, and velocity distribution for a number of realistic inlet shapes and tide conditions. An initial set of tests in a trapezoidal inlet was conducted between 1967 and 1970. However, in order that subsequent inlet models are more representative of real inlets, a number of "idealized" models representing various inlet morphological classes are being developed and tested. The effects of jetties and wave action on the hydraulics are included in the study.

   (2) Evaluation of state-of-the-art modeling techniques. The objectives of this part of the inlet hydraulics study are to determine the usefulness and reliability of existing physical and numerical modeling techniques in predicting the hydraulic characteristics of inlet/bay systems, and to determine whether simple tests, performed rapidly and economically, are useful in the evaluation of proposed inlet improvements. Masonboro Inlet, North Carolina, was selected as the prototype inlet which would be used along with hydraulic and numerical models in the evaluation of existing techniques. In September 1969 a complete set of hydraulic and bathymetric data was collected at Masonboro Inlet. Construction of the fixed-bed physical model was initiated in 1969, and extensive

tests have been performed since then. In addition, three existing numerical models were applied to predict the inlet's hydraulics. Extensive field data were collected at Masonboro Inlet in August 1974 for use in evaluating the capabilities of the physical and numerical models.

(3) Prototype inlet hydraulics. Field studies at a number of inlets are providing information on prototype inlet/bay tidal hydraulic relationships and the effects of friction, waves, tides, and inlet morphology on these relationships.

c. The basic objective of the inlet dynamics study is to investigate the interactions of tidal flow, inlet configuration, and wave action at tidal inlets as a guide to improvement of inlet channels and nearby shore protection works. The study is subdivided into four specific areas: model materials evaluation, movable-bed modeling evaluation, reanalysis of a previous inlet model study, and prototype inlet studies.

(1) Model materials evaluation. This evaluation was initiated in 1969 to provide data on the response of movable-bed model materials to waves and flow to allow selection of the optimum bed materials for inlet models.

(2) Movable-bed model evaluation. The objective of this study is to evaluate the state-of-the-art modeling techniques, in this case movable-bed inlet modeling. Since, in many cases, movable-bed modeling is the only tool available for predicting the response of an inlet to improvements, the capabilities and limitations of these models must be established.

(3) Reanalysis of an earlier inlet model study. In 1957, a report entitled, "Preliminary Report: Laboratory Study of the Effect of an Uncontrolled Inlet on the Adjacent Beaches," was published by the Beach Erosion Board (now CERC). A reanalysis of the original data is being performed to aid in planning of additional GITI efforts.

(4) Prototype dynamics. Field and office studies of a number of inlets are providing information on the effects of physical forces and artificial improvements on inlet morphology. Of particular importance are studies to define the mechanisms of natural sand bypassing at inlets, the

response of inlet navigation channels to dredging and natural forces, and the effects of inlets on adjacent beaches.

4. This appendix discusses the calibration, base tests, and predictive test of a numerical model of Masonboro Inlet, N. C., conducted as part of the evaluation of the state-of-the-art inlet modeling techniques. It presents the data necessary for a comparison of results of the physical and numerical models discussed in the basic report and in the following appendixes:

a. Appendix 1. R. A. Sager and W. C. Seabergh, "Fixed-Bed Hydraulic Model Results."

b. Appendix 2. F. D. Masch, R. J. Brandes, and J. U. Reagan, "Numerical Simulation of Hydrodynamics (WRE)" (In 2 Vols).

c. Appendix 4. C. J. Huval and G. L. Wintergerst, "Simplified Numerical (Lumped Parameter) Simulation."

ABSTRACT

The purpose of this project was to modify an existing two-dimensional numerical hydrodynamic model to be applied to Masonboro Inlet, North Carolina. The model essentially consists of a numerical solution to the time-dependent linearized shallow-water equation. One of the modifications was the incorporation of tidal flats which are important in the Masonboro Inlet system. The second was to devise a procedure for handling open water boundaries at which no boundary conditions were available. Using data for September 1969 supplied by CERC, the friction factor was adjusted to simulate the observed tide stage variation. Satisfactory verification was obtained against tidal stage variation and intratidal current phasing. The magnitude of the computed currents were consistently small but this is not thought to seriously delimit the model's utility.

ACKNOWLEDGMENTS

The authors are grateful to George H. Ward and A. D. Thomas of Tracor for fruitful discussions and valuable advice concerning this work.

TABLE OF CONTENTS

TABLE OF CONTENTS - Continued

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS - Continued

LIST OF ILLUSTRATIONS - Continued

LIST OF ILLUSTRATIONS - Continued

LIST OF TABLES

1.0 INTRODUCTION

The objective of this study was the adaptation of Tracor's two-dimensional hydraulic model to Masonboro Inlet, North Carolina, Fig. 1.1, in order to predict the water surface time history and current velocities from Masonboro Inlet for two hydrographic conditions. The project consisted of three main phases: a) adaptation of Tracor's model to Masonboro Inlet, b) adjustment of the model to allow reproduction of the prototype tides and currents of 12 September 1969, and c) prediction of tides and currents for the additional hydrographic conditions of the inlet for November 1964 and June 1967 using idealized mean and spring tides in the ocean.

Tracor's hydraulic model was initially developed in the Galveston Bay Project modeling program and is described in Espey *et al*. (1971). The model computes the dynamic vertical-mean water velocity components in the x- and y- directions for each grid cell in the bay system; these velocities are the hydrodynamic results of the major physical influences on the estuary or inlet, including tide, freshwater and wastewater inflows, bottom roughness, physiography and winds. The model has been used extensively on Galveston Bay as well as on Nueces Bay and Corpus Christi Bay.

Problems normally encountered in changing a hydraulic model from one set of physiographic conditions to another were aggravated in this study by the fact that a new flag field had to be developed to allow for flooding of the tidal flats and the unspecified boundary conditions at the ends of inlet channels. The development of a new flag field logic was made necessary because the logic originally contained in the program was not dynamic in nature, i.e. represented the estuary as a "fixed-bed" configuration. Program code also did not contain any provision for the unspecified boundary conditions, and this had to be added to the program.

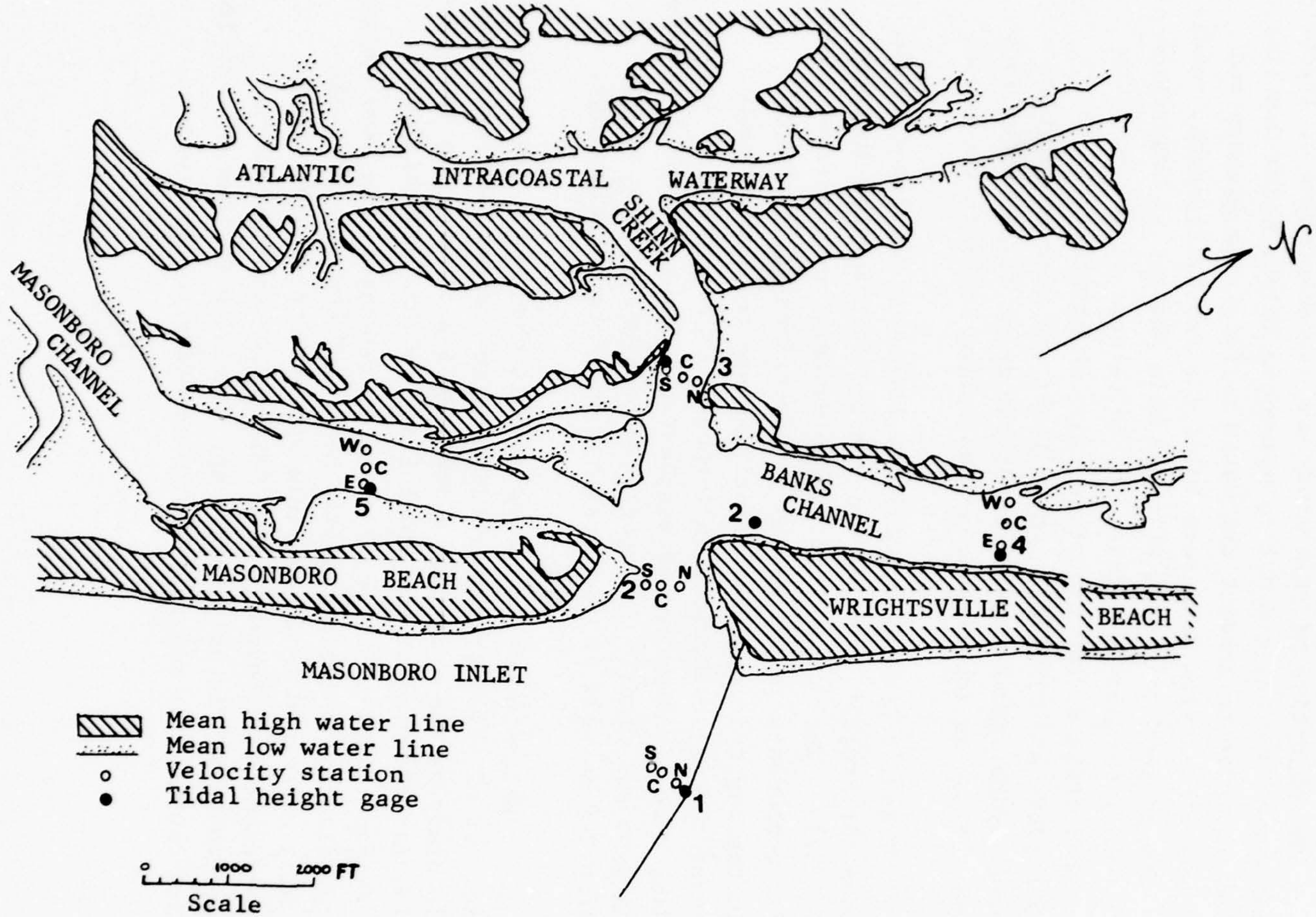


FIG. 1.1 MASONBORO INLET AREA

The basic theory and formulation of the numerical model is described in Section 2.0. The above modifications necessary for application to Masonboro Inlet are detailed in Section 3.0. Verification results and the computed results for the 1964 and 1967 cases are presented in Section 4.0. In Section 5.0 the FORTRAN programs are presented together with their corresponding flow charts and listings. The data deck arrangement and a discussion of each of the input parameters are given in Section 6.0.

2.0 THE HYDRAULIC MODEL

The basic mathematical hydraulic model used was the same as that described in Espey et al. (1971). However, extensive reprogramming was required to handle the tidal flats and the unspecified boundary conditions. The model considers the two horizontal dimensions representing the areal extent of the system using vertically integrated equations of motion. The model is dynamic; that is, it describes the time-varying aspects of the hydraulic phenomena.

2.1 The Basic Model

The differential equations comprising the basic mathematical model are referred to as the "linearized shallow water wave equations" or "the storm surge equations". The formulation employed in the Tracor hydraulic model follows closely that used by Reid and Bodine (1968) in their study of storm surges in Galveston Bay.

The vertically integrated equations of motion and continuity which form the basic mathematical hydraulic model are:

$$\frac{\partial Q_x}{\partial t} + gD\,\frac{\partial H}{\partial x} = KW^2\cos\theta - fQQ_xD^{-2} + Q_y 2\omega\sin\phi, \qquad (1)$$

$$\frac{\partial Q_y}{\partial t} + gD\,\frac{\partial H}{\partial y} = KW^2\sin\theta - fQQ_yD^{-2} - Q_x 2\omega\sin\phi, \qquad (2)$$

and

$$\frac{\partial H}{\partial t} + \frac{\partial Q_x}{\partial x} + \frac{\partial Q_y}{\partial y} = R, \qquad (3)$$

where x, y = horizontal Cartesian coordinates,

t = time,

Q_x, Q_y = vertically integrated x and y components of flow per unit width, respectively,

g = acceleration due to gravity,

D = total water depth,

H = water elevation with respect to MSL,

K = dimensionless wind stress coefficient from Wu (1969),

W = wind speed,

θ = angle between wind vector and the x-axis,

f = dimensionless friction coefficient after Reid and Bodine (1968),

$Q = \sqrt{Q_x{}^2+Q_y{}^2}$ = magnitude of flow per unit width,

ω = angular velocity of the earth,

ϕ = latitude,

and R = rainfall rate.

It will be noted that in equations (1) and (2) the advection of momentum terms are omitted; this omission is, in fact, the distinguishing feature of the storm-surge equations. The reasons for this omission are twofold: first, the nonlinear character of the advection of momentum terms renders the mathematics practically intractable, so that their omission results in a considerable mathematical simplification, and, second, these terms are not particularly important for the calculation of tidal hydraulics in shallow embayments or coastal areas. With regard to the former, the employment of numerical methods does not avoid the mathematical problems. In fact, the nonlinear terms de-stabilize the finite-difference computations even beyond that inherent in linear stability requirements. They further

introduce additional sources of numerical error in allowing nonlinear interaction between waves of various scales, thus permitting several types of "aliasing" phenomena, all of which can be disastrous computationally unless treated with considerable sophistication (see Kreiss and Oliger (1973)). With regard to the latter, use of the storm surge equations in shallow water tidal hydrodynamics is exemplified by Defant (1961), Dronkers (1964), Sverdrup et al. (1942), Neumann and Pierson (1966), Lazanoff (1971), Masch et al. (1969), and many others. Espey et al. (1971) note that even though this approximation may be generally justified, there can be local regions where it fails, e.g. in the vicinity of high-volume inflow sources such as tributary mouths or power plant cooling water discharges. In these localities, the nonlinear terms must be retained. Fortunately, there do not seem to be any regions within Masonboro Inlet where this is required.

To implement the above mathematical model (the equations), a means of relating the model to a physical system is required. This is accomplished by superimposing a lattice or grid system over the area to be modeled. The spacing within the lattice is determined by the degree of spatial resolution which is required by the particular problem. Required resolution may range from a few hundred feet to over a mile depending upon the phenomenon to be investigated. The boundaries of the physical system must be represented within the cell structure by the "walls" of the cells, therefore requiring a square grid representation of the physical system, as demonstrated in Figure 2.1. In the example, some of the detail of the real boundary is lost with the cell structure representation. A smaller grid spacing could have been used; the finer the grid, the better the representation. However, there is generally a trade-off between more accurate spatial

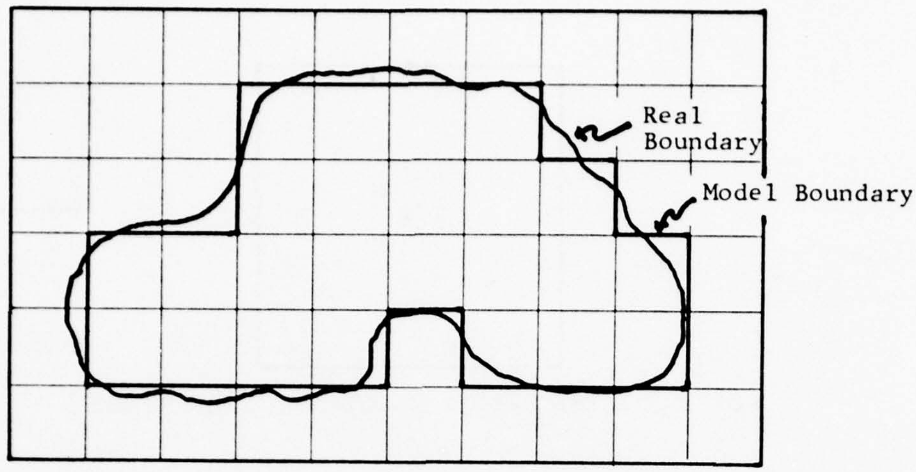


FIG. 2.1 CELL STRUCTURE EXAMPLE

representation and cost, since a finer grid requires more computer time to solve the necessary equations.

The grid system serves as the structure upon which the mathematical model and the physical system are joined. The various parameters are assigned values and positions within the cell structure. Figure 2.2 shows the positions within a grid where variables are defined in this model.

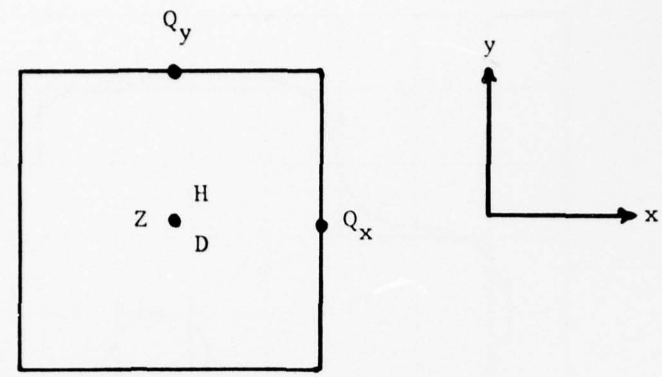


FIG. 2.2 DEFINITION OF HYDRAULIC PARAMETERS

At the center of each cell a depth Z of the bottom of the bay with respect to mean sea level (MSL) is assigned to represent the topography of the estuary. Again, the spatial detail of the model is affected by the cell size, as the depth assigned to each cell must represent an average over the corresponding area of the bay. As the water level H relative to some datum changes dynamically (while the model is being exercised) the instantaneous water depth D is obtained by algebraically adding H to the constant Z. Also note in Figure 2.2 that the components of flow per unit width (Q_x and Q_y) are defined at the right side and top side of each cell, respectively.

2.2 Boundary Conditions

Numerous boundary conditions are necessary in the hydraulic model to reproduce physical conditions. Among these are the simulation of land boundaries, open water boundaries, and submerged barriers and reefs. In general, the boundary conditions are implemented by assigning prescribed values to the components of flow per unit width (Q_x and Q_y) or the water level (H) relative to the datum. Land-water boundaries in the

model are handled in a straightforward manner by simply setting the component of flow per unit width normal to the boundary equal to zero; e.g., $Q_x = 0$ or $Q_y = 0$, depending upon the boundary's orientation. Open water boundaries are required to input a driving tide at the opening of the bay to the ocean. This requires that the water elevation with respect to the datum be specified. This tidal record is obtained from gauge records of the United States Corps of Engineers (USCE). Given this time history of water level, the flow per unit width is calculated by a modification of equations (1) and (2).

$$\frac{\partial Q_x}{\partial t} + gD\frac{\partial H}{\partial x} = -fQQ_xD^{-2} \tag{4}$$

$$\frac{\partial Q_y}{\partial t} + gD\frac{\partial H}{\partial y} = -fQQ_yD^{-2} \tag{5}$$

Generally, the open boundary is oriented so that either x or y is constant, hence only one of (4) and (5) is employed.

An initial distribution of H within the model domain is specified arbitrarily (often taken to be constant for convenience), and the observed tidal stage variation is specified at the boundary. From this initial condition, $\partial H/\partial x$ and/or $\partial H/\partial y$ in (4) and (5) may be computed, and the time evolution of Q_x , Q_y and H determined therefrom. The errors introduced by the initial distribution of H induce a transient response in the system hydraulics, but this transient soon propagates out of the system (a consequence of the hyperbolic nature of equations (1) and (2)), the precise time required depending upon its physiographic and hydraulic features.

Other open water boundaries may occur within an inlet where no tide records may be available, for example, where the

channels in Masonboro Inlet intersect the grid boundary. When this occurs there appear to be two recourses to resort to, 1) make up a tidal record or obtain the tidal record from other considerations, or 2) using physical constraints, extrapolate the boundary condition. The second method follows a suggestion of Bodine (private communication) and was used in this application to Masonboro Inlet. The variable extrapolated is the change in water level H. A simple linear extrapolation is used; i.e.,

$$\Delta H_i = 2\Delta H_{i-1} - \Delta H_{i-2}$$

where the Δ indicates the change between time steps and i is the cell index. The additional physical constraints are also used (superscript indicates relative time step)

<u>Ebb Tide</u>	<u>Flood Tide</u>
$H_i^{\nu+1} \leq H_i^{\nu}$	$H_i^{\nu+1} \geq H_i^{\nu}$
$H_i^{\nu+1} \geq H_{i-1}^{\nu+1}$	$H_i^{\nu+1} \leq H_{i-1}^{\nu+1}$

When one reflects that the hydraulic behavior at these open boundaries is in reality determined by the interaction between the tidal hydraulics of the inlet area itself (which is within the bounds of the computational model) and that of the adjacent waterways and tidal inlets both upcoast and downcoast (which are not within the boundaries of the model), then it becomes apparent that this approach may not be adequate. However, these boundaries can be handled with physical fealty only by either acquiring empirical hydraulic data to be input as boundary conditions, or by extending the model bounds to encompass the entirety of the coastal hydraulic system. The former is impossible since the

required data are not available, and the latter is manifestly beyond the scope of the present project. Accordingly, the above approximation was elected.

Boundary conditions for submerged barriers and reefs closely follow those used by Reid and Bodine (1968). The discharge over a submerged barrier is taken as that for a submerged weir, specifically

$$Q_x \text{ (or } Q_y\text{)} = \pm C_s D_b \sqrt{g|H_1 - H_2|} \, , \tag{6}$$

where C_s is the barrier discharge coefficient, D_b is the water depth over the barrier, and $|H_1 - H_2|$ is the difference in water level on either side of the barrier. Q_x or Q_y is computed depending upon the orientation of the barrier with respect to the cell structure and the algebraic sign is assigned to correspond to flow directed toward the low-head side of the barrier.

2.3 Model Implementation

Final implementation of the hydraulic model requires that the differential equations (Equations (1), (2), and (3)) be written in an explicit finite-difference form using one-sided temporal and centered spatial differences with the appropriate spatial averaging. These finite-difference equations can then be advanced through time to provide a time history of water surface behavior and components of flow per unit width at all grid points described in the model grid. These compose the principal output of the hydraulic model, because from these the tidal history at any point in the bay and the spatial and temporal distribution of flow and horizontal velocity can be determined.

In Figure 2.2, the points within the cell structure at which the primary variable is defined are depicted in general. When implementing the finite-difference solution, a scheme for

spatially indexing the variables and associating the indices with the cell structure is required, as well as a determination of the relationship of the variables with one another in time. The spatial orientation is shown in Figure 2.3. (Note that within each individual cell, the scheme of Figure 2.2 is maintained.) The first index on each variable (i) increases in the positive x direction, and the second (j) in the positive y direction. The lower left cell in the structure is therefore cell (1,1). (For simplicity, all variables are shown only on cell (i, j) in Figure 2.3, but their location remains the same at any other cell.) Wind stress, friction, and Coriolis force terms are computed at locations consistent with the particular flow (Q_x or Q_y) that is affected.

The temporal scheme for assigning the variables is very important to accuracy and simplicity of the model formulation. When advancing the finite-difference equations through time, it is important that all the terms involved in the equation be computed for the same instant in time. To accomplish this, the variable H , and associated variables, D , Z , and R , and the variables Q_x and Q_y , are assigned positions in time one-half time increment (Δt) apart. That is to say Q_x and Q_y are computed at even half-time steps, $t = n\Delta t$, and H is computed at odd half-time steps, $t = (n + \frac{1}{2})\Delta t$, where n takes on increasing integer values. By adopting this convention, the time associated with the quantity $(Q_x^{(n+1)\Delta t} - Q_x^{n\Delta t})$ is the same as that of $H^{(n+\frac{1}{2})\Delta t}$ (the superscripts referring to the time) and likewise $(H^{(n+3/2)\Delta t} - H^{(n+\frac{1}{2})\Delta t})$ is at the same time as $Q_x^{(n+1)\Delta t}$. Reference to the finite-difference equations below will show that this convention yields equivalent time throughout each equation.

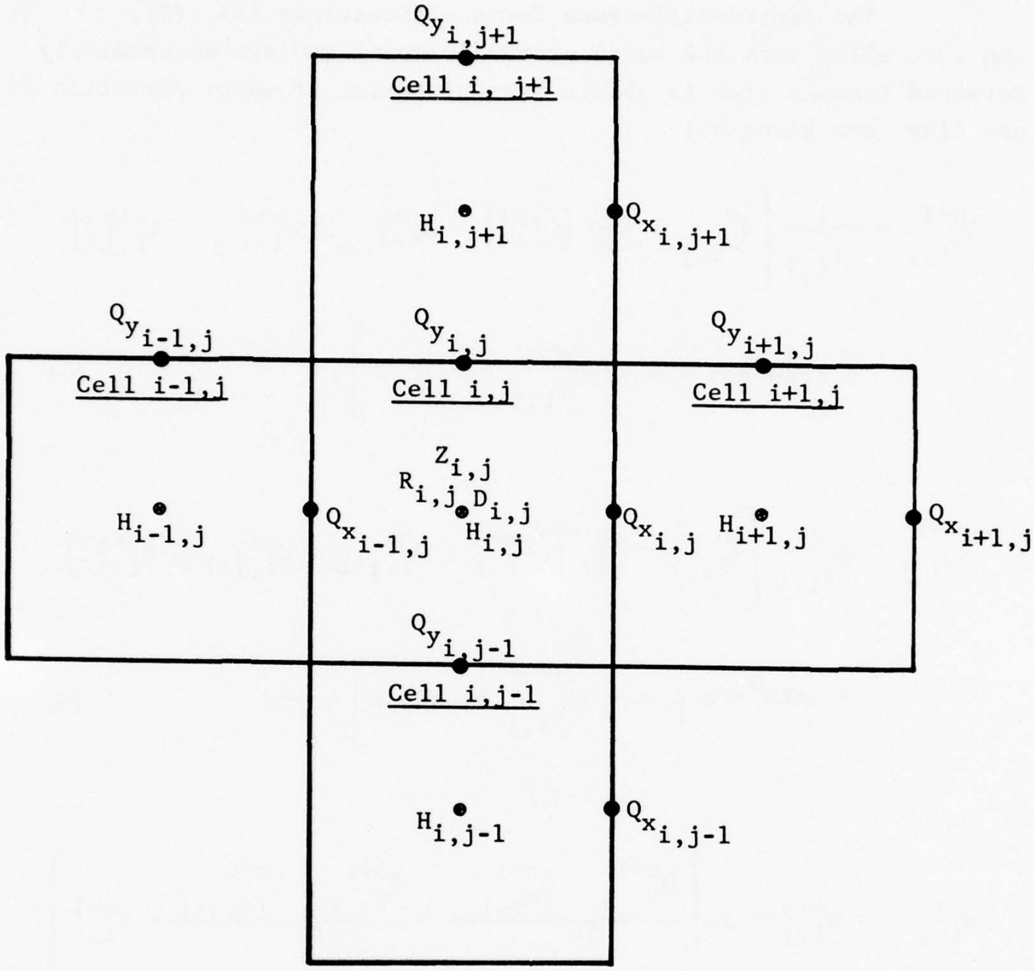


FIG. 2.3 CONVENTION EMPLOYED IN DESIGNATION OF HYDRAULIC VARIABLES.

The finite-difference forms of Equations (1), (2), and (3), which form the basic hydraulic model and are successively advanced through time to obtain time histories of water elevation and flow, are given by:

$$Q_{x_{i,j}}^{n+1} = \frac{1}{G_{1_{i,j}}} \left[Q_{x_{i,j}}^{n} - \frac{g\Delta t}{2\Delta x} \left[\left(D_{i,j}^{n+\frac{1}{2}} + D_{i+1,j}^{n+\frac{1}{2}} \right) \left(H_{i+1,j}^{n+\frac{1}{2}} - H_{i,j}^{n+\frac{1}{2}} \right) \right] \right.$$

$$\left. + \Delta t K W^2 \cos\theta + \Delta t \, \overline{Q_{y_{i,j}}^{n}} \, 2\omega \sin\phi \right], \tag{7}$$

$$Q_{y_{i,j}}^{n+1} = \frac{1}{G_{2_{i,j}}} \left[Q_{y_{i,j}}^{n} - \frac{g\Delta t}{2\Delta y} \left[\left(D_{i,j}^{n+\frac{1}{2}} + D_{i,j+1}^{n+\frac{1}{2}} \right) \left(H_{i,j+1}^{n+\frac{1}{2}} - H_{i,j}^{n+\frac{1}{2}} \right) \right] \right.$$

$$\left. + \Delta t K W^2 \sin\theta - \Delta t \, \overline{Q_{x_{i,j}}^{n}} \, 2\omega \sin\phi \right], \text{ and} \tag{8}$$

$$H_{i,j}^{n+3/2} = H_{i,j}^{n+\frac{1}{2}} + \Delta t \left[\frac{Q_{x_{i,j}}^{n+1} - Q_{x_{i-1,j}}^{n+1}}{\Delta x} + \frac{Q_{y_{i,j}}^{n+1} - Q_{y_{i,j-1}}^{n+1}}{\Delta y} + R_{i,j}^{n+1} \right]. \tag{9}$$

where $$\overline{Q_{x_{i,j}}} = \frac{Q_{x_{i-1,j}} + Q_{x_{i,j}} + Q_{x_{i-1,j+1}} + Q_{x_{i,j+1}}}{4} \tag{10}$$

and $$\overline{Q_{y_{i,j}}} = \frac{Q_{y_{i,j-1}} + Q_{y_{i,j}} + Q_{y_{i+1,j-1}} + Q_{y_{i+1,j}}}{4} \tag{11}$$

Also, the effect of friction has been incorporated into the terms G_1 and G_2 after Reid and Bodine (1968).

$$G_{1_{i,j}} = 1 + f\Delta t \left[\frac{\left\{ \left(Q_{x_{i,j}} \right)^2 + \left(\overline{Q_{y_{i,j}}} \right)^2 \right\}^{\frac{1}{2}}}{\left(\frac{D_{i,j} + D_{i+1,j}}{2} \right)^2} \right] \tag{12}$$

and

$$G_{2_{i,j}} = 1 + f\Delta t \left[\frac{\left\{ \left(Q_{y_{i,j}} \right)^2 + \left(\overline{Q_{x_{i,j}}} \right)^2 \right\}^{\frac{1}{2}}}{\left(\frac{D_{i,j} + D_{i,j+1}}{2} \right)^2} \right] . \tag{13}$$

It is also preferable to incorporate the effect of submerged barriers into an equivalent friction factor, specifically:

$$G_{1_{i,j}} = 1 + \frac{\Delta t}{2\Delta y}\left[\frac{(D_{i+1,j} + D_{i,j})\ |Q_{x_{i,j}}|}{(C_s D_b)^2}\right] \tag{14}$$

and

$$G_{2_{i,j}} = 1 + \frac{\Delta t}{2\Delta x}\left[\frac{(D_{i,j+1} + D_{i,j})\ |Q_{y_{i,j}}|}{(C_s D_b)^2}\right], \tag{15}$$

depending upon the orientation of the barrier within the cell structure.

The size of the time increment (Δt) which can be used to advance the finite difference equations through time is dictated by numerical stability. Platzman (1958) presents the following stability requirement:

$$\Delta t \leq \frac{\Delta s}{\sqrt{2g\ D_{max}}} \tag{16}$$

where Δs is the grid spacing in both directions, i.e., $\Delta s = \Delta x = \Delta y$, and D_{max} is the maximum water depth expected during the run.

After the grid structure has been determined, the hydraulic model is operated as follows:

1) Initial values for all variables are set. These initial conditions may be from a previous model operation or some set of unrealistic conditions which must be overcome by operating the model through an initial transient stage;

2) Using the values of Q_x and Q_y at time 0 and H at time $\Delta t/2$, the values of Q_x and Q_y at Δt and H at $3\Delta t/2$ are computed using the finite-difference forms presented

above. Boundary conditions for submerged barriers, land boundaries, freshwater inflow, and tidal input are specified as required;

3) Step 2 is repeated as time is advanced, computing Q_x and Q_y at $(n+1)\Delta t$ and H at $(n+3/2)\Delta t$ as n increases;

4) In the notation (i, j) the i index corresponds to the x axis or direction and the j index corresponds to the y axis or direction. Normally the origin would be in the lower lefthand corner and the i index would increase from left to right and the j index from bottom to top. In this notation the cells with the same i index would be a column and those with the same j index would be a row. However, in a two-dimensional computer array the subscript (1,1) is normally taken to be in the upper lefthand corner, with the first subscript indicating the row and the second subscript the column. Therefore, in transferring the grid indices from the actual grid to the array in the computer the grid is rotated 90° in a clockwise direction and the rows of the grid become column in the computer array and the column of grid become rows in the computer array. Therefore, for the purpose of clarity hereafter when a row or column is referred to it is as defined in the actual grid system, not as defined in the computer array. That is, a row is parallel to the x axis and a column is parallel to the y axis.

The order in which the cells are processed is as follows, assuming an n by m grid. The program starts with the first cell in row one (cell [1,1], the bottom lefthand cell) and works its way across the row to the last cell in row one (cell [1,m]). The program then moves up to the first cell in the second row (cell [2,1]) and works its way across the second row to cell [2,m]. After the program has finished the second row, it moves to the third. It continues this process

until it has reached the last cell in the last row, (cell [n,m]) at which point the computations are complete for that time step. In treating any given cell, the right face is always treated before the top face.

3.0 MODEL DEVELOPMENT

Additional development of the existing computer program was necessary to apply the two-dimensional hydraulic model to Masonboro Inlet. The areas that needed further development were 1) the procedure for handling unspecified boundary conditions, and 2) the revision for the flag field for tidal flats.

3.1 Unspecified Boundary Conditions

In the grid system adopted (Δx , Δy = 300 feet), the three arms of the inlet intersect the grid boundaries (points A, B, and C in Fig. 3.1). At these intersections there are open water boundaries where some form of boundary condition is required. There are two kinds of boundary conditions that can be used in an open water boundary: 1) the flow rate can be specified, or 2) the water elevation can be specified. In the dynamic model these boundary conditions will be time varying. However, in the data available to this project there were no measurements that could be used as a boundary condition at these points. Therefore, some means had to be developed to handle these boundaries.

As noted in the preceding chapter there are basically two approaches open to handle this problem in the present model. These are 1) a set of water level records can be fabricated to use as the boundary conditions, or 2) a means of extrapolating a variable to use as a boundary condition needs to be found. The latter approach was chosen because the former could be approached in two ways, neither of which are considered satisfactory. One way is to prescribe <u>a priori</u> the tidal record, however this introduces subjective error. The other is to implement a model encompassing a larger area and using the results from that model to specify the tides at the boundaries of the present model. It was felt, however, that this approach deviated too much from the original project plan.

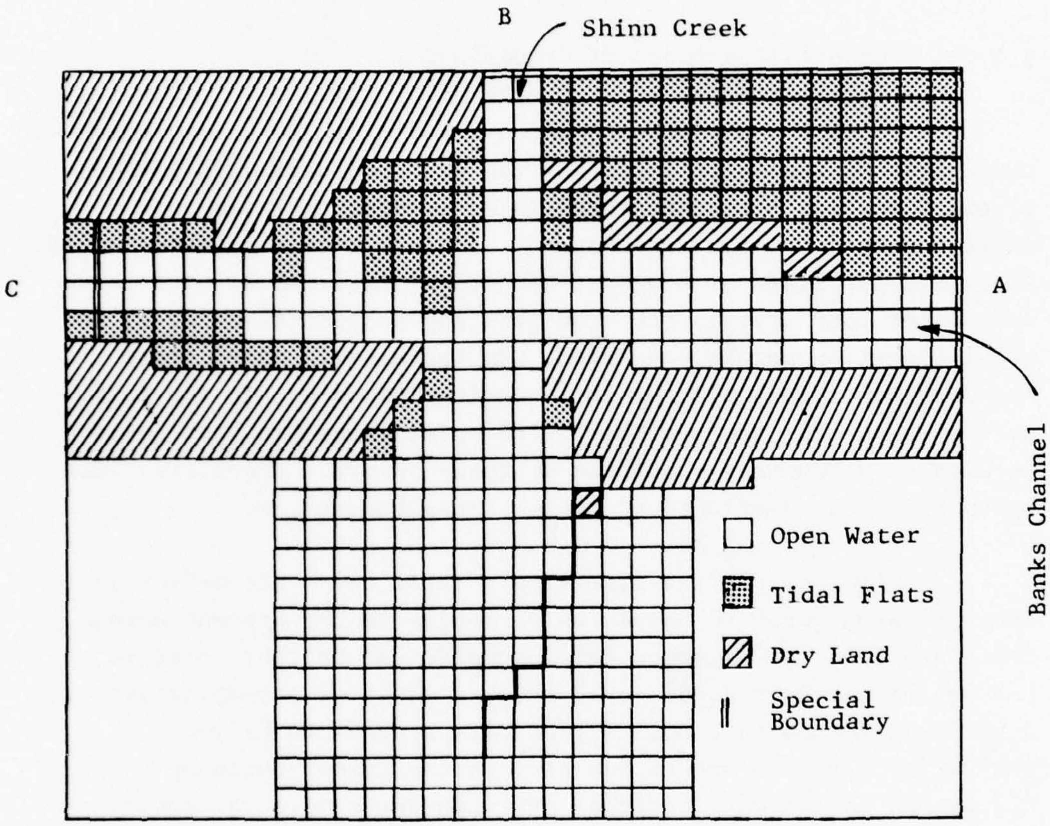
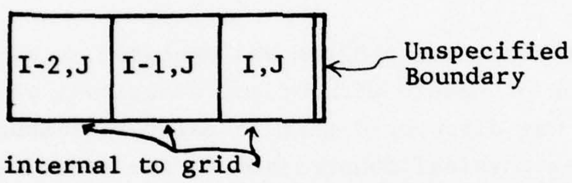


FIG. 3.1 GRID SYSTEM FOR SEPTEMBER 1969 CASE
(TOTAL FLOOD PLANE INCLUDED)
(Grid size = 300 feet)

Once the second approach had been elected, there remained the problem of determining which variable to extrapolate. The variables that are available are the water level H , the change in water level ΔH , and the flow rate Q . It was felt that the flow rate was not a suitable variable for extrapolation because it is too dependent on the bottom topography and on other flows into the cell. It was decided that ΔH would be the better variable to extrapolate.

A linear extrapolation procedure was used. The change in $H_{I,J}$ is found by the equation



$$\Delta H^{\nu+1}_{I,J} = 2\left(H^{\nu+1}_{I-1,J} - H^{\nu}_{H-1,J}\right) - \left(H^{\nu+1}_{I-2,J} - H^{\nu}_{I-2,J}\right)$$

where superscript ν indicates the time step. Similar equations are used if one of the other cell walls has an unspecified boundary condition. The new water level at cell (I,J) is found by adding $\Delta H^{\nu+1}_{I,J}$ to the old water level, i.e.

$$H^{\nu+1}_{I,J} = H^{\nu}_{I,J} + \Delta H^{\nu+1}_{I,J} .$$

The newly found change in water level $\Delta H^{\nu+1}_{I,J}$ is then used in equation (9), page 13, to find the flow associated with the cell wall having the unspecified boundary condition. In the example given above, the resulting equation would be

$$Q^{\nu+1}_{x_{I,J}} = Q^{\nu+1}_{x_{I-1,J}} + Q^{\nu+1}_{y_{I,J-1}} - Q^{\nu+1}_{y_{I,J}} - \frac{\Delta x}{\Delta T}\left(\Delta H^{\nu+1}_{I,J}\right) .$$

A requirement that is a result of this extrapolation procedure is that the cells (I-1,J) and (I-2,J) must be flooded while cell (I,J) is flooded, otherwise erroneous results would occur. This requirement is important in tidal flats which have unspecified boundary conditions as do the tidal flats to the west of Banks Channel and north of Shinn Creek, see Fig. 3.1.

At first the program was operated using only the extrapolation procedure without any additional physical constraints. However, it was discovered that to maintain computational stability the following physical constraints on the water level in the cell for which the extrapolation was performed had to be included. The equations given correspond to the above example.

For ebb tide:

1. The extrapolated water level must be less than the old water level:

$$H^{\nu+1}_{I,J} < H^{\nu}_{I,J} .$$

2. The extrapolated water level must be greater than the water level in the cell from which the extrapolation was performed:

$$H^{\nu+1}_{I,J} > H^{\nu+1}_{I-1,J} .$$

For flood tide:

1. The extrapolated water level must be greater than the old water level:

$$H^{\nu+1}_{I,J} > H^{\nu}_{I,J} .$$

2. The extrapolated water level must be less than the water level in the cell from which the extrapolation was performed:

$$H^{\nu+1}_{I,J} < H^{\nu+1}_{I-1,J} .$$

3.2 Flag Field Revision

The flag field is the means by which the program knows what calculations to perform. It indicates the nature of the cell wall, i.e. whether there is flow through it, if it is a land boundary, etc. The flag field that was originally employed in Tracor's hydraulic model was a fixed flag field; that is, once specified it did not change as the program ran. Therefore, the program could not have cells in it which were drained or inundated. Where there were extensive tidal flats the program could not adequately represent the area. Since there are several tidal flats in the Masonboro Inlet, the flag field had to be revised to handle the cells in these areas. Furthermore, the flag for the unspecified boundary also had to be implemented. The old flag field system did not lend itself to being efficiently modified to become a dynamic flag field, therefore, a completely new flag field system was developed.

The flag system developed in this project is based upon setting bits in the computer word. Depending upon which bits are set, various branches in the program flow are selected. The last (least significant) 9 bits of the computer word are used. In the following discussion bit positions are given from the right,

that is, the rightmost bit is bit number 1. Also, that a bit is set means that it has the value one (1). If it is not set, it is zero (0).

The significance of the bit positions is given in Table 3.1. When bits 7 or 8 are set, a no-flow condition is indicated for the corresponding walls.

TABLE 3.1

CORRESPONDENCE OF BITS IN COMPUTER WORD
AND WHAT THEY CORRESPOND TO

Bits	Indicate
1 to 3	Condition of right wall
4 to 6	Condition of top wall
7	Flow or no-flow - left wall
8	Flow or no-flow - bottom wall
9	Cell permanently flooded
>9	Not used

Bit 9 is set when a cell is always flooded. The key to bits 1 to 3 and 4 to 6 is given in Table 3.2. A listing of all possible flag combinations is given in Appendix A along with their decimal equivalent. This information is necessary for decoding a flag field print-out.

For further notes on how to specify the initial flags read by the program see Section 6.8 on the flag cards.

TABLE 3.2

KEY TO FLAG BITS 1 TO 6 FOR RIGHT AND TOP WALLS OF CELL IN GRID SYSTEM

Bits	Top Wall 6 5 4	Right Wall 3 2 1	Decimal Equivalent	Octal Equivalent*	Wall Status
		0	0	0	Free flow
		1	1	1	Barrier
		1 0	2	2	Tidal input
		1 1	3	3	Special boundary
		1 0 0	4	4	External tidal input
		1 0 1	5	5	Variable boundary
		1 1 0	6	6	External+
		1 1 1	7	7	Permanent boundary
	0		0	0	Free flow
	1		8	1	Barrier
	1 0		16	2	Tidal input
	1 1		24	3	Special boundary
	1 0 0		32	4	External tidal input
	1 0 1		40	5	Variable boundary
	1 1 0		48	6	External+
	1 1 1		56	7	Permanent boundary

*Octal equivalent of three digits composing flag for that wall.

+All cell walls that are not involved in calculations, i.e., cell walls internal to a land mass.

4.0 MODEL VERIFICATION, RESULTS AND CONCLUSIONS

4.1 Verification

The model was verified using the September 1969 data supplied. Verification was to consist of adjusting the model to reproduce as closely as possible the measured tides and velocities provided in the data. The locations of the sampling stations are shown in Fig. 4.1. The actual process of verification consists of operating the model with one set of friction factors, checking the results, and then rerunning using a different set of friction factors. This process is repeated until the best reproduction of tides and velocities is obtained.

It was originally planned to include the flood plain north of Shinn Creek and to the west of Banks Channel as shown in Fig. 3.1. However, numerical problems were encountered while the flood plain was draining. Oscillations were set up whenever a cell emptied, which caused the cells to alternately flood and drain. Another problem encountered on the flood plains was the occasional draining of a cell while a shallower cell adjacent to it was still flooded. This would cause the cell to dry up then flood again. This may have been associated with the aforementioned oscillations or may have been a separate numerical problem arising from the flow from the shallower cell to the deeper cell not being large enough. An attempt was made to rectify these problems by adjusting the friction coefficients in the flood plain. However, this did not correct the difficulty, and it was judged that the problem did not merit the time which would be required to pursue other approaches. Therefore, because it was not the flood plain itself but its boundaries with Shinn Creek and Banks Channel that were important in the computations, this flood plain was eliminated from the model and only its boundaries with the two creeks modeled. The tidal flats at the intersection of Masonboro Channel and Shinn Creek are left in the computations.

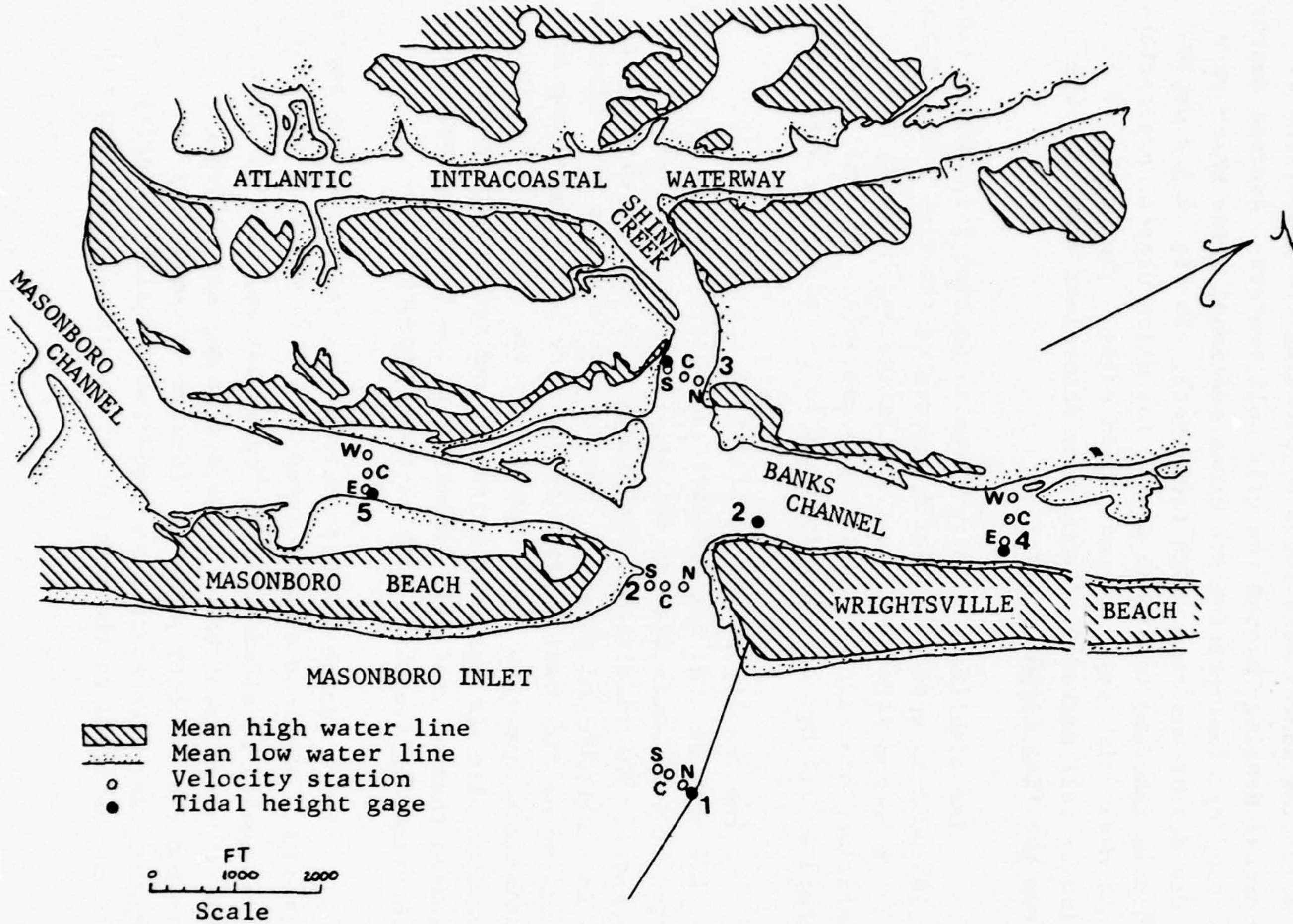


FIG. 4.1 LOCATION OF MASONBORO INLET SAMPLING STATIONS

The grid structure that was used for verification is shown in Fig. 4.2. The corresponding depths used are shown in Fig. 4.3. The depths were found by plotting cross sections of the channel passing through the grid cell centers. Average depths were found by planimetering the cross sectional areas which were then divided by the cell width (300 feet). In Fig. 4.3 a depth of 99.9 is inserted to signal a cell for which direct computations are not made, viz. dry land and unused cells. The status of a particular cell marked 99.9 should be determined from Fig. 4.2 or from the flag field proper.

The stability criteria given in Section 2 (equation (16), page 16) puts an upper limit of 6 seconds on the time step. Therefore, a 6 second time step was tried initially; however, the computations proved to be numerically unstable. In order to maintain stability, the time step had to be reduced to 5 seconds.

The results of the verification are presented in Figs. 4.4 to 4.24. In the verification runs the model was started at 0400 hours with an initial tidal level of .951 feet throughout. The tidal stage verification is considered adequate both in amplitude and phasing. The current velocity time histories calculated by the model appear satisfactory in zero-crossing and in phasing in general. Their amplitudes are low, however, in comparison with the measured currents, and sensitivity tests indicated that no realistic variation of the friction coefficients could yield amplitudes as high as those measured.

There can be several contributing factors to the amplitude difference. The most obvious seems to lie in the fact that the system modeled is effectively an "open" system, thus the distribution of water level through the system may not uniquely determine the velocity pattern. Insofar as continuity is concerned, any divergence-free (two-dimensionally) velocity field may be added to the model solution without violating (3)

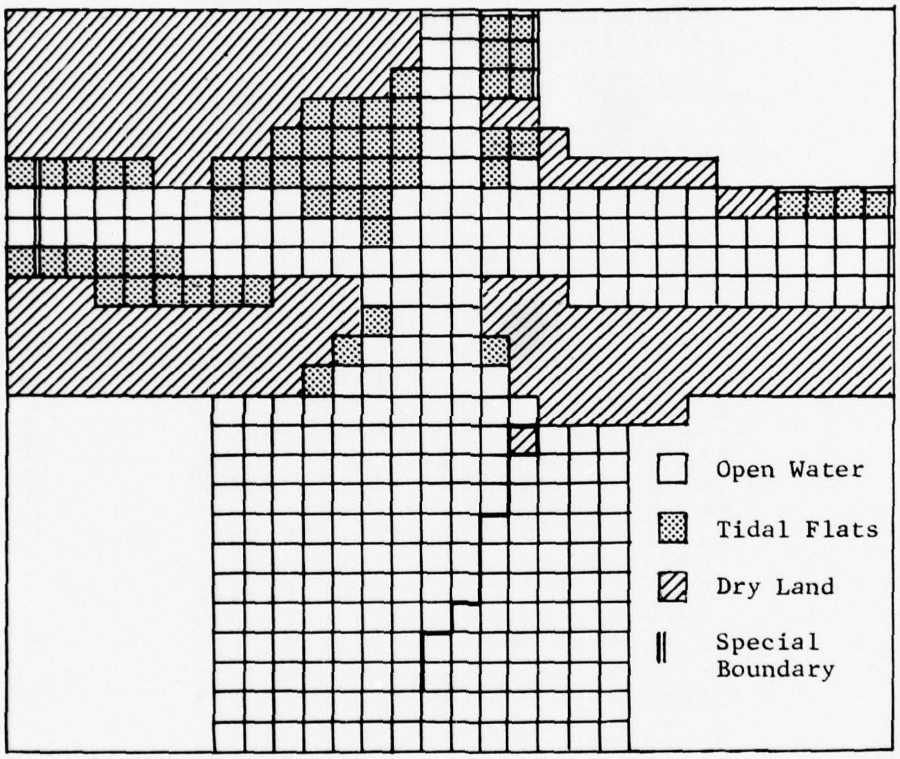
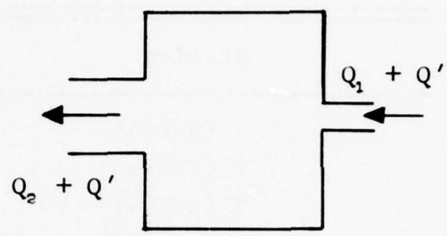


FIG. 4.2 GRID SYSTEM FOR SEPTEMBER 1969 CASE

I/J	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-1.8	3.0	4.0	-3.0	99.9	99.9	99.9	99.9	99.9
2	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-1.8	3.3	4.0	-3.0	99.9	99.9	99.9	99.9	99.9
3	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-1.7	3.3	4.6	-2.9	99.9	99.9	99.9	99.9	99.9
4	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-2.3	-1.3	4.6	4.1	-2.8	99.9	99.9	99.9	99.9	99.9
5	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-2.3	-1.0	5.1	3.8	-2.8	99.9	99.9	99.9	99.9	99.9
6	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-2.0	.8	6.1	2.2	99.9	99.9	99.9	99.9	99.9	99.9
7	99.9	18.0	13.0	11.0	11.0	10.0	9.0	9.0	9.0	8.0	6.0	3.0	99.9	99.9	99.9	-1.5	2.0	6.3	1.5	99.9	99.9	99.9	99.9	99.9	99.9
8	12.0	12.0	11.0	10.0	10.0	9.0	8.5	8.5	8.0	7.0	6.0	4.0	99.9	99.9	99.9	-1.5	2.9	6.4	-1.5	-1.2	99.9	99.9	99.9	99.9	99.9
9	11.0	11.0	9.5	8.0	9.0	8.0	8.0	8.0	7.0	7.0	5.0	4.0	99.9	99.9	99.9	-1.5	3.1	6.0	.5	-1.0	99.9	99.9	99.9	99.9	99.9
10	10.0	9.5	8.0	8.0	8.0	6.0	7.0	8.0	9.0	7.0	6.0	5.0	99.9	99.9	99.9	99.9	2.7	5.6	.5	-1.5	-1.0	99.9	99.9	99.9	99.9
11	10.0	8.5	5.5	7.5	6.5	6.0	5.5	6.0	5.0	5.0	5.0	6.0	-.5	99.9	99.9	99.9	4.3	4.3	-1.0	-1.5	-.7	-1.5	99.9	99.9	99.9
12	14.0	10.0	5.6	4.2	4.0	3.0	3.0	4.0	4.0	3.0	2.0	4.0	5.0	-1.0	99.9	99.9	5.6	2.1	-1.5	-.8	-.6	-2.5	99.9	99.9	99.9
13	18.0	18.0	10.6	7.2	3.4	1.0	1.0	1.0	1.5	1.0	1.5	2.0	3.0	.5	-1.0	3.0	5.0	-1.0	-1.5	-.7	-.5	-3.5	99.9	99.9	99.9
14	23.0	21.0	14.3	18.0	10.7	6.1	2.4	1.0	1.5	3.1	6.4	6.7	5.1	5.8	8.7	11.8	5.4	3.1	.9	-1.0	-.4	-2.0	-3.5	99.9	99.9
15	23.0	21.0	14.0	10.0	19.3	16.7	15.8	10.6	8.0	9.8	12.5	17.8	27.0	26.0	19.0	23.5	20.7	12.7	11.7	11.7	11.9	8.9	5.9	3.8	3.0
16	28.0	23.0	16.0	13.0	10.0	8.5	14.4	16.4	10.4	17.6	16.8	14.9	13.7	9.9	8.2	8.2	14.5	13.2	7.0	8.9	8.2	12.2	18.0	16.2	18.0
17	28.0	23.0	18.5	15.0	14.0	11.5	9.0	6.0	7.5	9.6	7.1	4.6	1.5	-1.0	99.9	99.9	9.7	5.5	6.1	-.5	-1.0	99.9	-1.4	-1.5	-1.6
18	28.0	23.0	20.0	19.0	18.0	14.0	11.0	7.0	3.0	2.0	99.9	.2	99.9	99.9	99.9	99.9	9.5	2.2	5.1	1.3	-2.0	99.9	-1.9	-1.9	-2.0
19	28.0	28.0	23.0	23.0	20.0	17.0	14.0	9.0	7.5	2.0	2.0	99.9	99.9	99.9	99.9	99.9	10.7	1.9	5.3	99.9	99.9	99.9	99.9	99.9	99.9
20	29.0	28.0	28.0	24.0	20.5	17.0	15.0	13.0	11.0	5.0	2.0	99.9	99.9	99.9	99.9	1.9	11.4	1.9	4.8	99.9	99.9	99.9	99.9	99.9	99.9
21	34.0	32.0	32.0	27.0	21.0	19.0	15.5	13.5	13.0	9.0	3.0	99.9	99.9	99.9	99.9	1.7	11.6	3.8	3.6	99.9	99.9	99.9	99.9	99.9	99.9
22	99.9	34.0	32.0	28.0	22.0	20.0	17.0	15.0	14.0	11.5	4.0	99.9	99.9	99.9	99.9	1.4	11.8	8.7	2.7	99.9	99.9	99.9	99.9	99.9	99.9
23	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	3.5	12.3	12.4	1.2	99.9	99.9	99.9	99.9	99.9	99.9
24	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	5.6	13.2	11.9	.5	99.9	99.9	99.9	99.9	99.9	99.9
25	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	6.4	15.8	10.8	99.9	99.9	99.9	99.9	99.9	99.9	99.9
26	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	5.6	17.3	6.0	99.9	99.9	99.9	99.9	99.9	99.9	99.9
27	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	6.3	17.6	6.0	-.5	99.9	99.9	99.9	99.9	99.9	99.9
28	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	8.0	17.0	8.0	-.5	99.9	99.9	99.9	99.9	99.9	99.9
29	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	8.0	13.5	1.5	-.5	99.9	99.9	99.9	99.9	99.9	99.9
30	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	8.0	13.0	2.0	-.6	99.9	99.9	99.9	99.9	99.9	99.9

FIG. 4.3 DEPTH FIELD FOR SEPTEMBER 1969 CASE

(for R = 0) if the boundary conditions are given in terms of H alone. In contrast, for a "closed" system Q is specified somewhere (Q = 0 for a head of tide or Q = discharge for an inflowing somewhere (Q = 0 for a head of tide or Q = discharge for an inflowing river) so the velocity is uniquely determined. A simple-minded analogy is a two-port system with a variable H which is specified at each opening. The value of Q′ in the figure obviously has no effect upon the distribution or time variation of H. Conversely, knowledge of H alone (viz. boundary conditions or continuity) cannot permit complete inference of the flow, since Q′ cannot be determined. If one boundary condition is reformulated in terms of discharge rather than H, then this problem disappears. The analogy to Masonboro Inlet is apparent (although the dynamic equations (1) and (2) will delimit considerably the form the indeterminate divergence-free velocity component can assume). It would appear that either the model must be extended to encompass the entirety of the inlet system, or the boundary condition at the ocean must be in terms of discharge rather than H. It must be emphasized that this is conjectural, and deserves a thorough theoretical treatment. It does however suggest why the same model applied to other shallow embayments gives much more satisfactory current magnitudes, in that all of the other applications did not involve an "open" boundary (other than that at the ocean).



An additional possible explanation for this discrepancy in velocities lies in the measured data themselves. In studying the empirical data of Figs. 4.6 et seq, one notices that in general there is a net discharge over the tidal cycle; this is particularly prominent at Stations 1 and 2. Approximate tidal-mean currents

TABLE 4.1

APPROXIMATE TIDAL-MEAN CURRENTS,
FROM MEASUREMENTS OF 12 SEPTEMBER 1969
(Positive denotes flood, negative ebb)

Station	Net Velocity (fps)
1 (north)	-1.0
1 (center)	-1.1
1 (south)	-0.8
1 (mean)	-1.0
2 (north)	+0.8
2 (center)	-0.8
2 (south)	-1.0
2 (mean)	-0.3
3 (north)	-0.2
3 (center)	-0.2
3 (south)	+0.5 (?)
3 (mean)	0.0
4 (west)	+0.1
4 (center)	-0.1
4 (east)	-0.3
4 (mean)	-0.1
5 (west)	0.0
5 (center)	-0.3
5 (east)	-0.1
5 (mean)	-0.1

from the time histories of Figs. 4.6 _et seq_ are tabulated in Table 4.1. The station "means" in this table are simple arithmetic means of the three lines to give an indication of the tidal net flow; no effort has been made to weight the lines by their tidal depth. It can be seen from this table that the net discharge into the inlet region is practically nil, as indicated by the mean currents at Stations 3, 4 and 5 (see Fig. 4.1). Nevertheless there is a dramatic tidal-mean discharge at each line of both Stations 1 and 2. At Station 1, in the main channel, the net flow is unidirectional and amounts to nearly 10,000 cfs _net_. At Station 2, the tidal-mean circulation appears to describe a gyre with inward flow on the north side and outward flow on the south, with an intensity on the order of 3. x 10^3 cfs and a net discharge across the section on the order of 2. x 10^3 cfs.

It is clear that the processes incorporated in the model computations (viz., head gradient driven by tidal variation at mouth, frictional resistance, coriolis acceleration) are incapable of yielding circulations such as these. A precise agreement between model results and data can be expected only if all of the processes operating in the real world are represented in the model, which does not seem to be the case here. As there are no supporting observations available, we can only speculate as to the cause of the departure between the observed situation and the idealized model calculations. The most probable causes are thought to be one or a combination of the following:

(1) The inlet system for the September 1969 data was in a transient state, i.e. not in the tidal-mean equilibrium assumed in the model computations. In particular, at the start of the data run (0700 EST 12 September), there was a surfeit of water within the system, thus driving a net discharge over the succeeding period. Although such a situation is not theoretically precluded

by the model formulation, its simulation requires much more complete initial conditions, viz. point-by-point specification of water level throughout the system at the preceding slack current.

(2) The system was responding throughout the test period to systematic accelerations not included in the model computations. Two prime candidates are:

(i) wind stress, producing wind-driven gyres in semi-enclosed areas. In order to include this effect in the model calculations, representative observations of wind speed and direction within the Inlet and coastal area are required. These were not available, however, so the wind speed was arbitrarily set to zero.

(ii) longshore currents, whose interaction with the jetty produces a tidal-mean eddy. To include this effect, at the very minimum tidal stage variations are required both upcoast and downcoast from the Inlet entrance.

Factors such as these will not only influence the tidal-mean velocities but the intratidal velocities as well. The importance of such factors in the September 1969 data is strongly suggested by the tidal-mean currents of Table 4.1. We infer therefore that it is likely that the intratidal time histories of Figs. 4.6 _et seq_. are similarly influenced, and for this reason are not particularly alarmed at the discrepancy between the measured tidal current amplitudes and the model calculations. More precise comparison of model results and measurements requires a more complete data set.

4.2 Model Runs

The November 1964 and June 1967 cases were each run twice using two driving tides, 1) a mean tide and 2) a spring tide. These tides are presented in Fig. 4.25.

The hydrographic survey for the November 1964 case is shown in Fig. 4.26 and the corresponding grid and depths used are given in Figs. 4.27 and 4.28. The predicted tides and velocities are shown in Figs. 4.29 to 4.33 for the mean tide, and Figs. 4.34 to 4.38 for the spring tide.

The hydrographic survey for the June 1967 case is shown in Fig. 4.39 and the corresponding grid and depths in Figs. 4.40 and 4.41. The predicted tides and velocities are shown in Figs. 4.42 to 4.46 for the mean tide, and Figs. 4.47 to 4.51 for the spring tide.

The predicted tidal prisms are given in Table 4.2 for all the runs including the verification run.

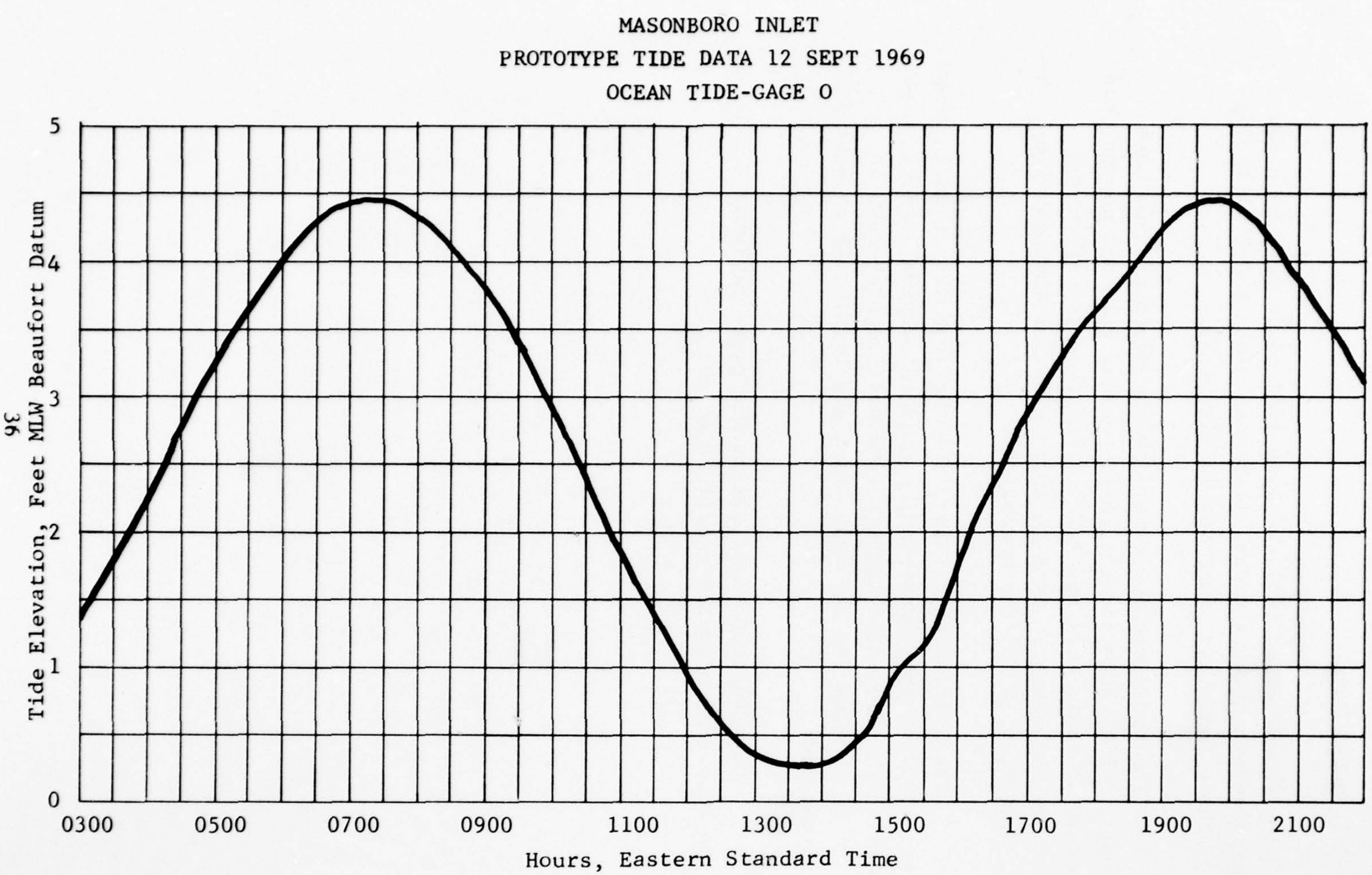


FIG. 4.4 DRIVING TIDE USED IN VERIFICATION MODEL

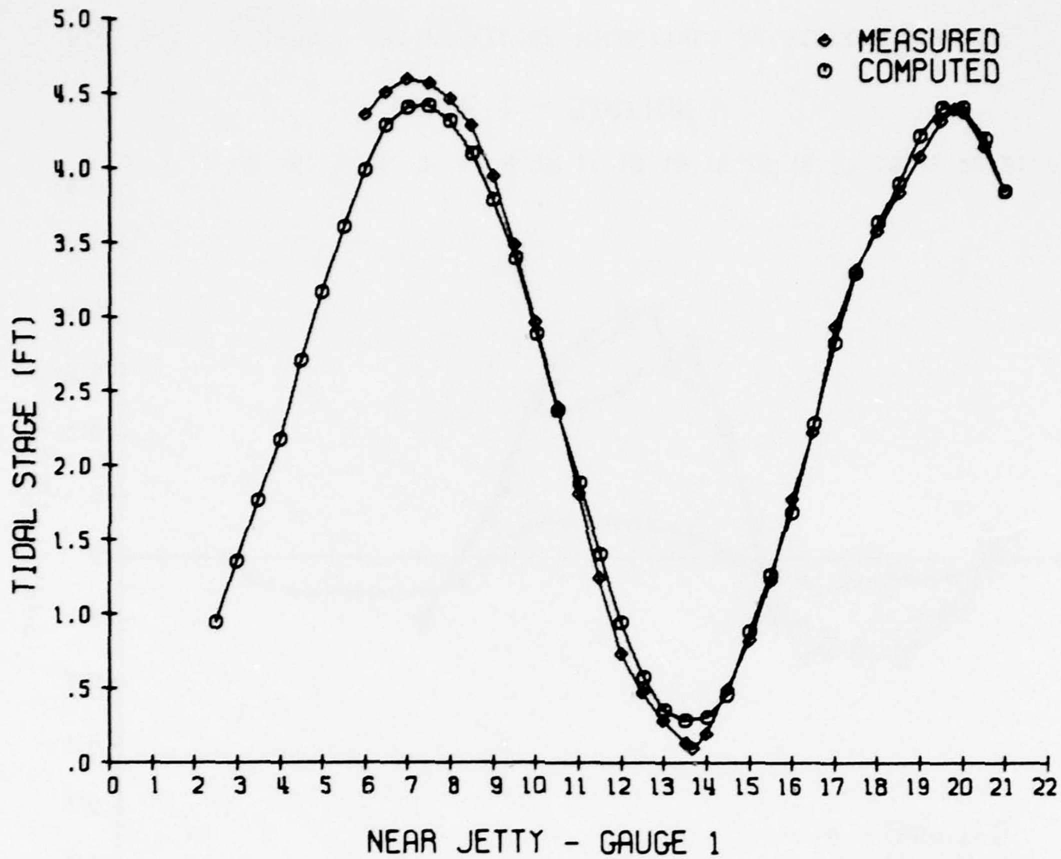


FIG. 4.5 MEASURED AND PREDICTED TIDE FOR STATION 1, VERIFICATION RUN

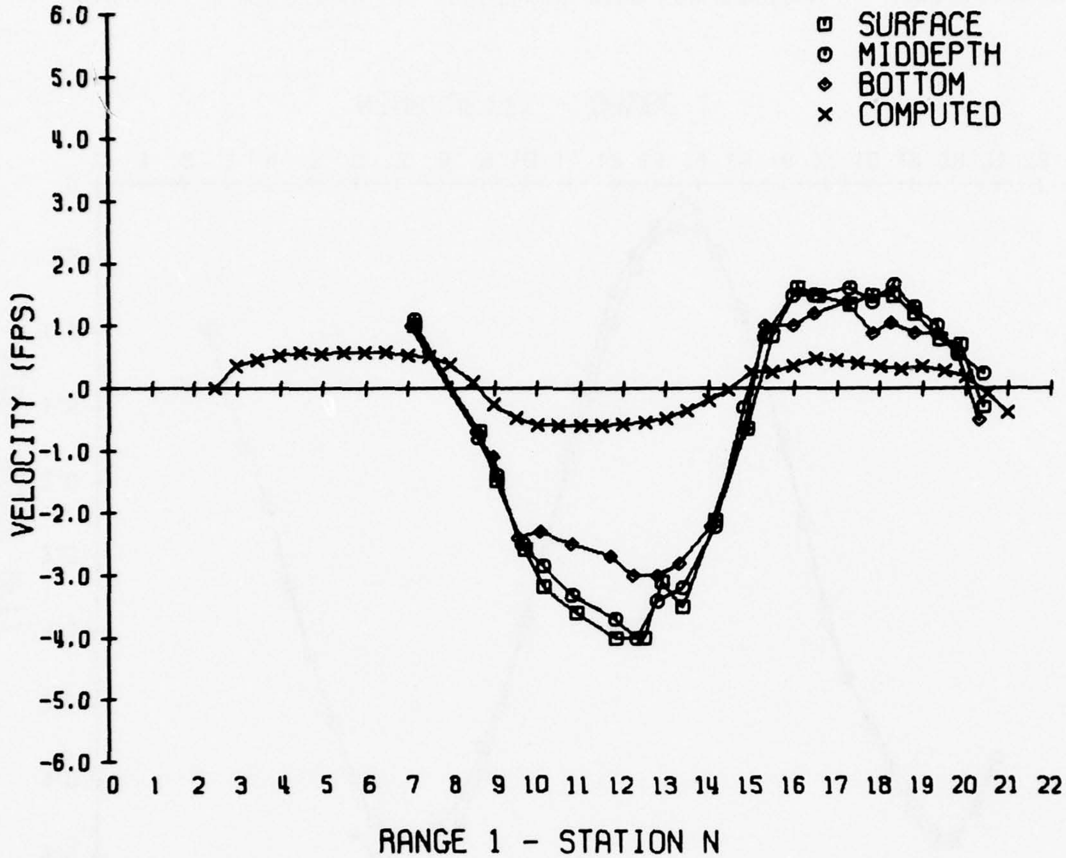


FIG. 4.6 MEASURED AND PREDICTED VELOCITIES AT STATION 1 NORTH, VERIFICATION RUN

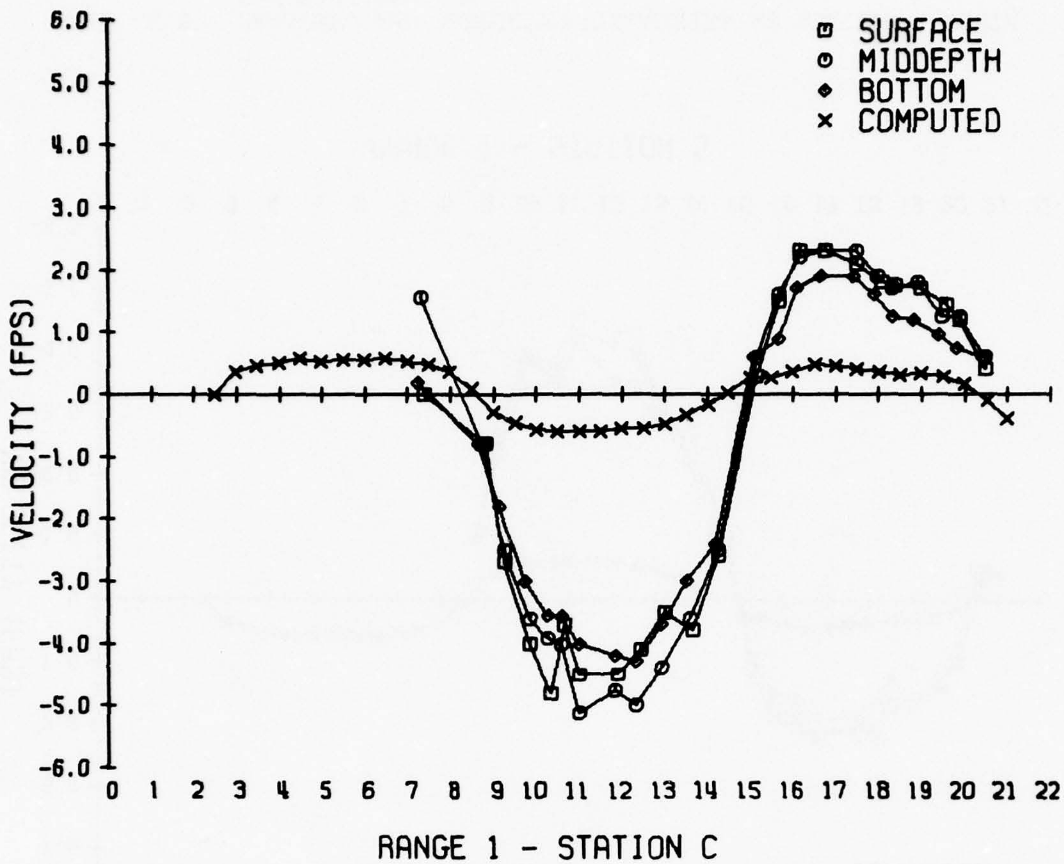


FIG. 4.7 MEASURED AND PREDICTED VELOCITIES AT STATION 1 CENTER, VERIFICATION RUN

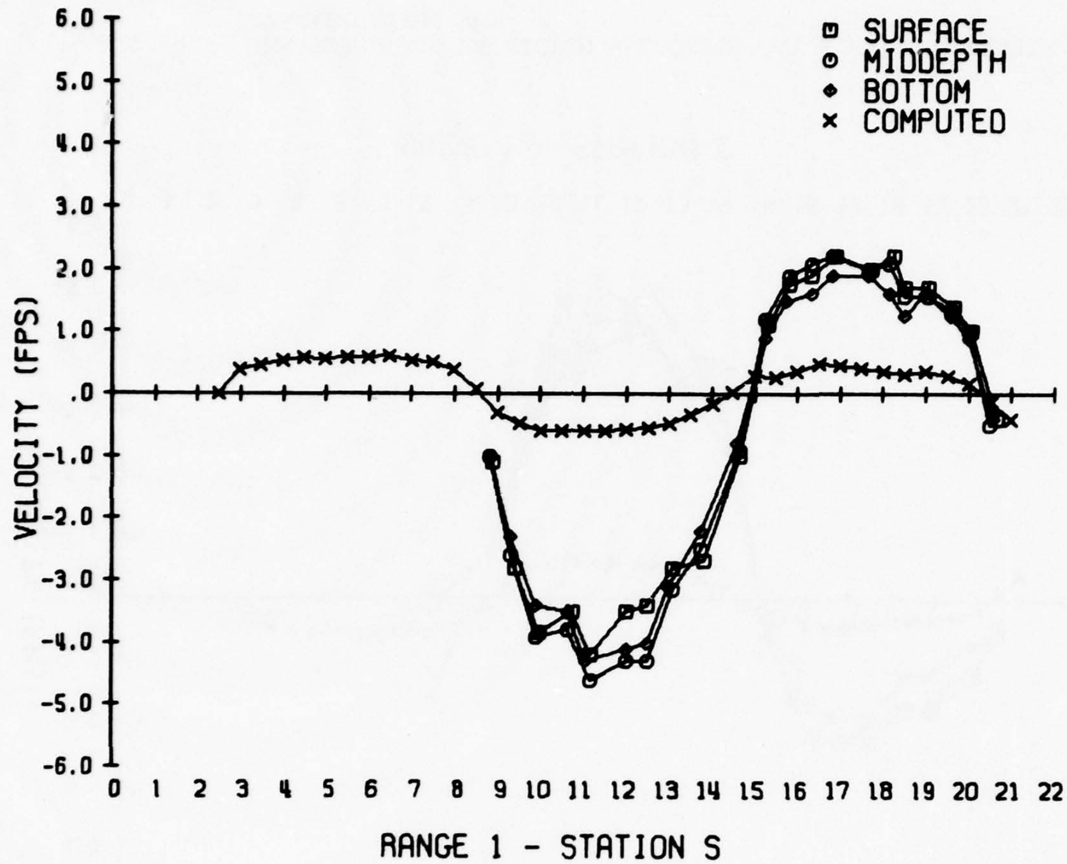


FIG. 4.8 MEASURED AND PREDICTED VELOCITIES AT STATION 1 SOUTH, VERIFICATION RUN

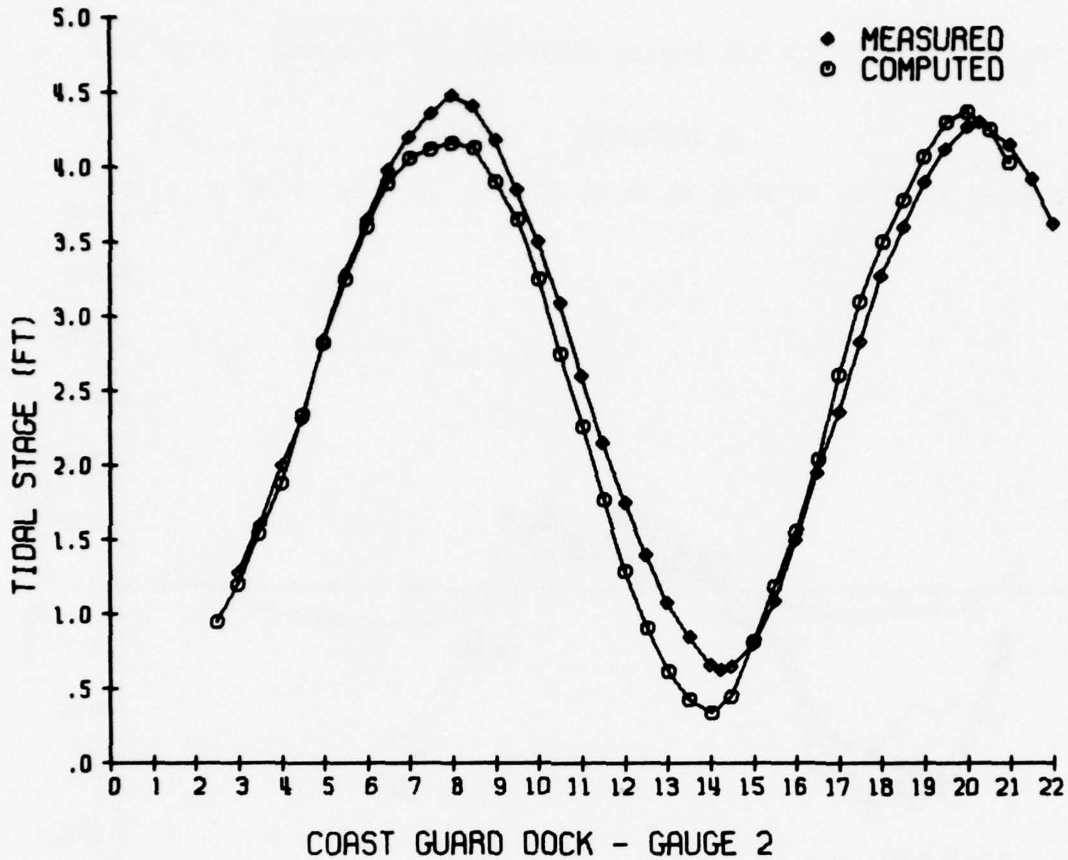


FIG. 4.9 MEASURED AND PREDICTED TIDE FOR STATION 2, VERIFICATION RUN

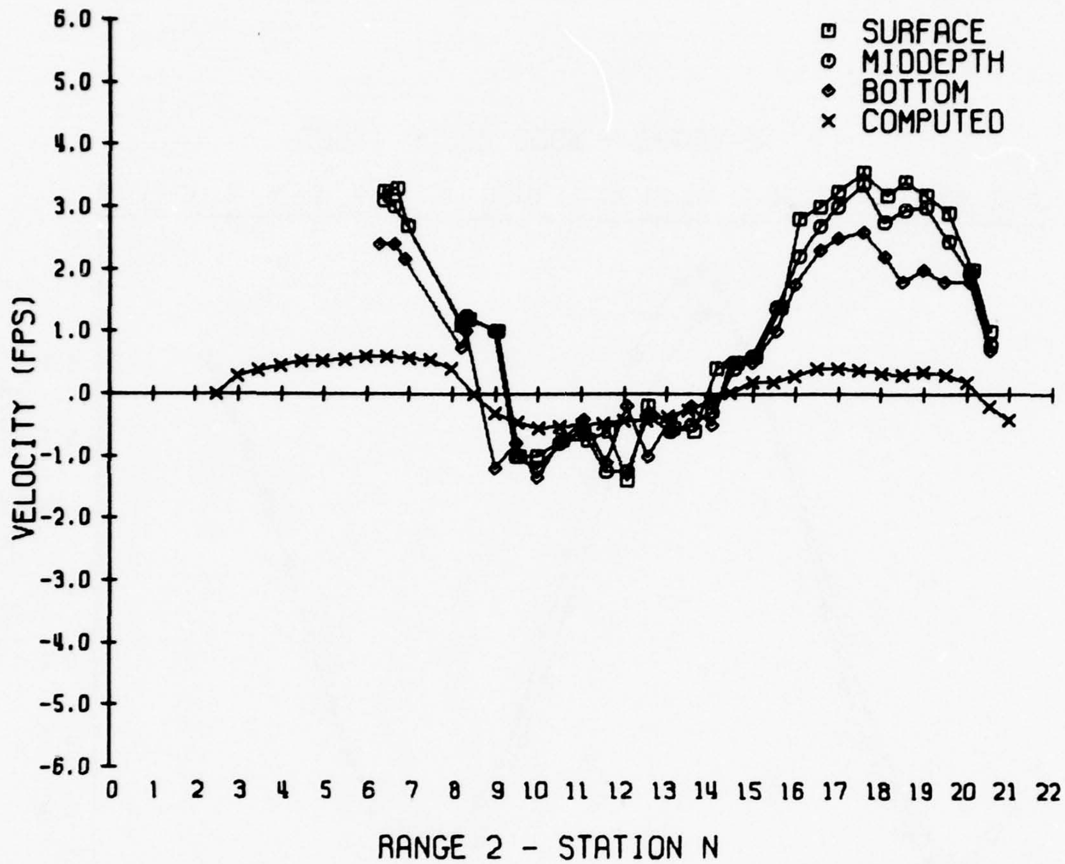


FIG. 4.10 MEASURED AND PREDICTED VELOCITIES AT STATION 2 NORTH, VERIFICATION RUN

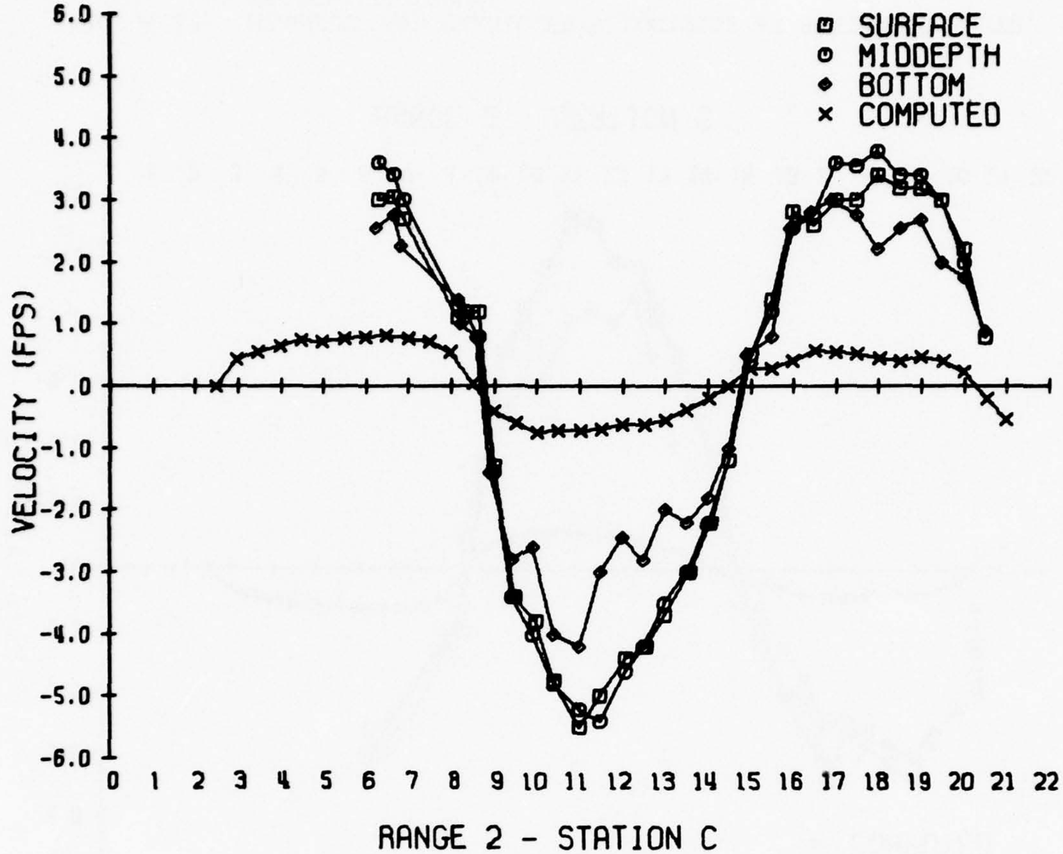


FIG. 4.11 MEASURED AND PREDICTED VELOCITIES AT STATION 2 CENTER, VERIFICATION RUN

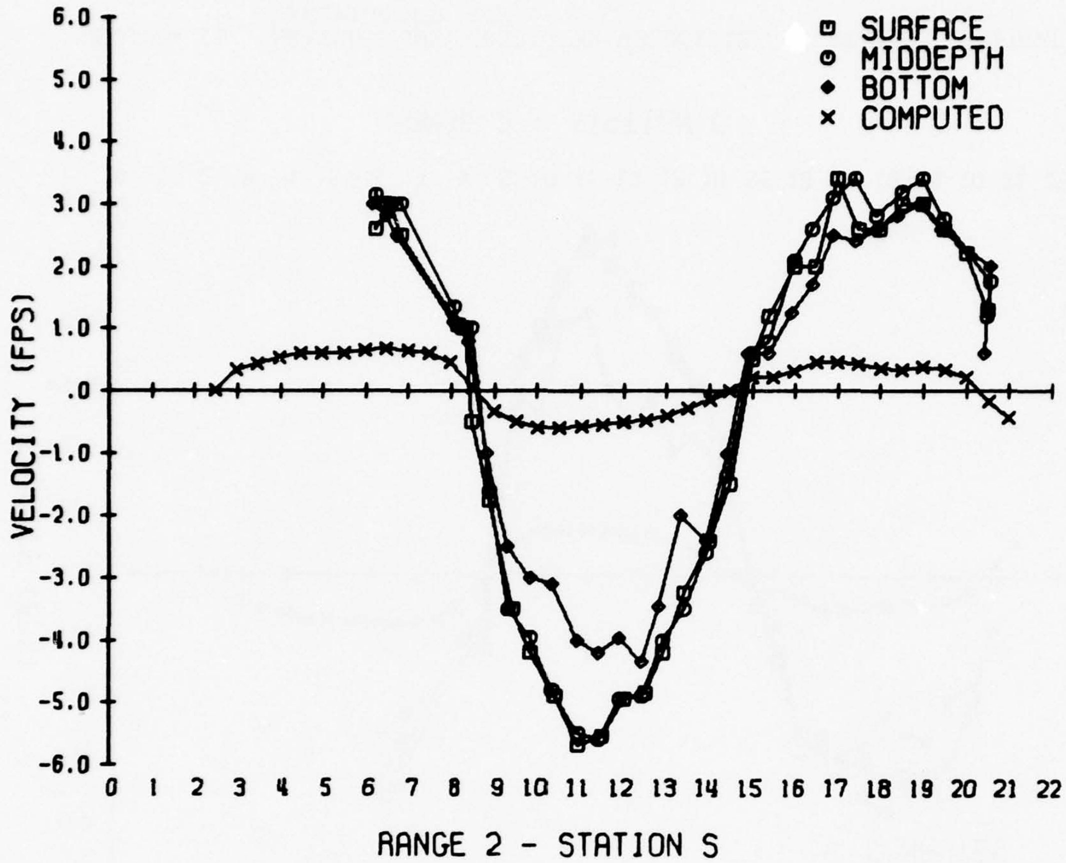


FIG. 4.12 MEASURED AND PREDICTED VELOCITIES AT STATION 2 SOUTH, VERIFICATION RUN

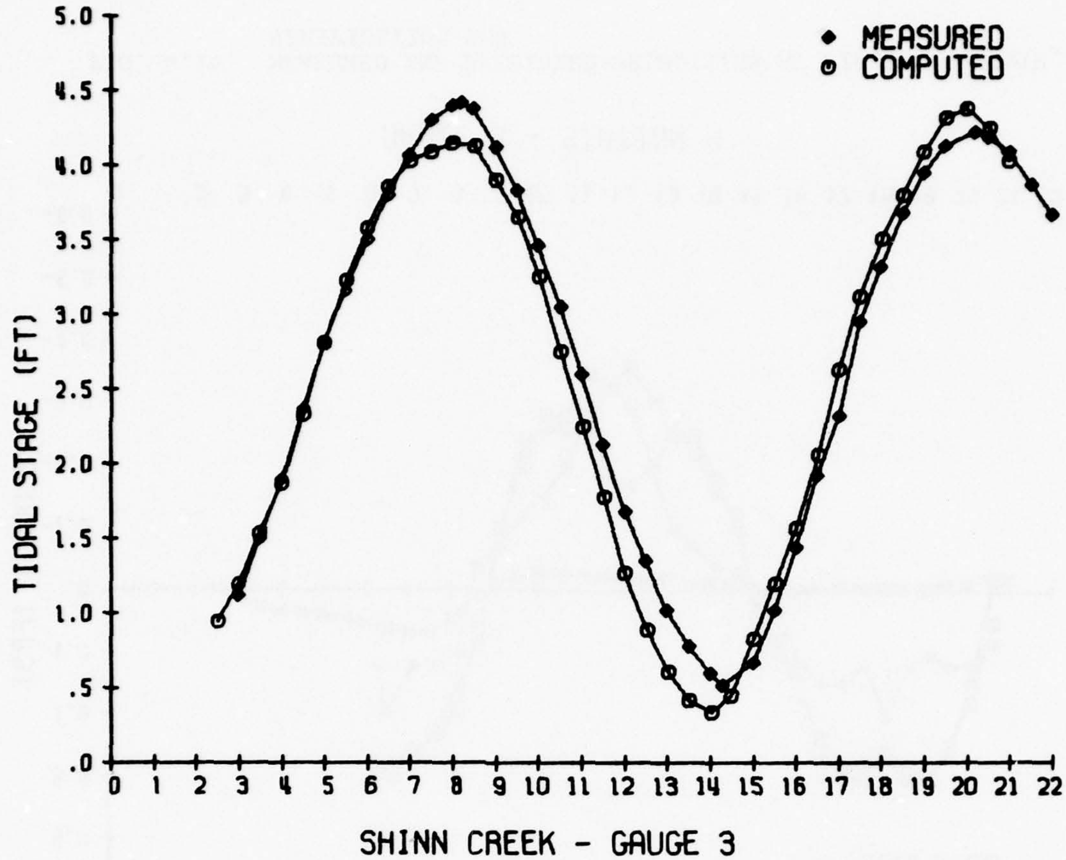


FIG. 4.13 MEASURED AND PREDICTED TIDE FOR STATION 3, VERIFICATION RUN

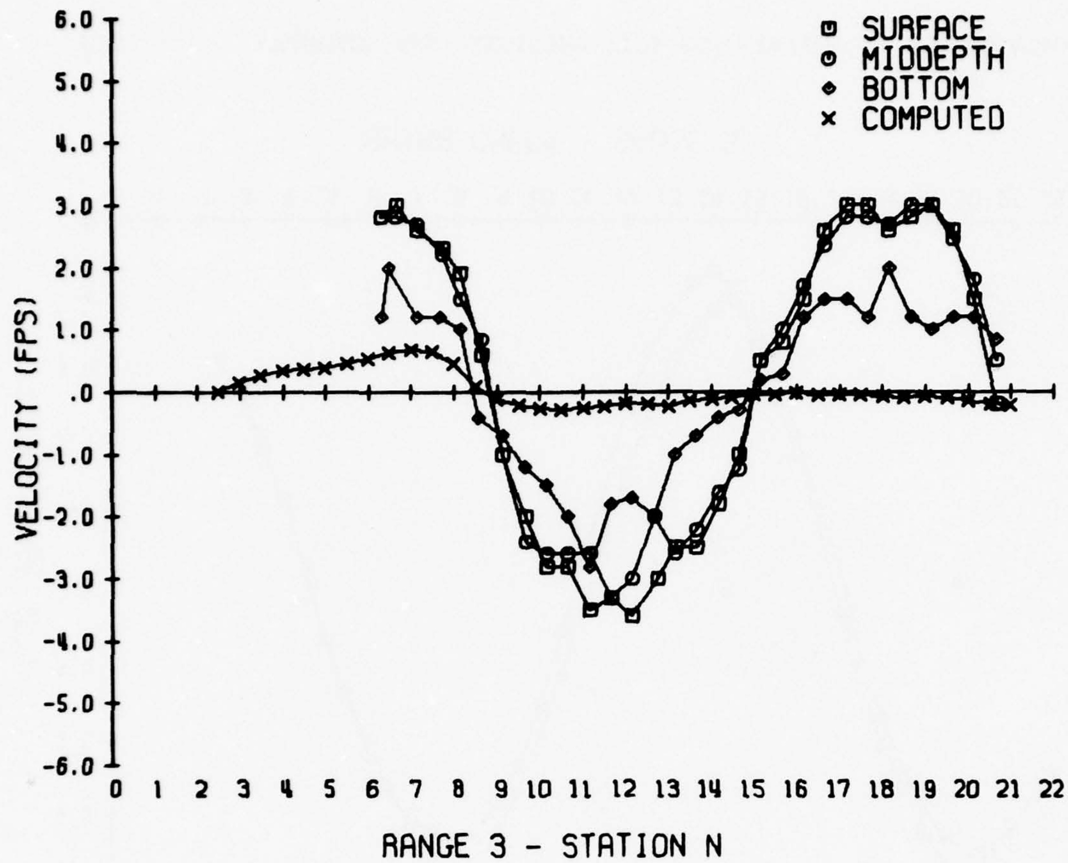


FIG. 4.14 MEASURED AND PREDICTED VELOCITIES AT STATION 3 NORTH, VERIFICATION RUN

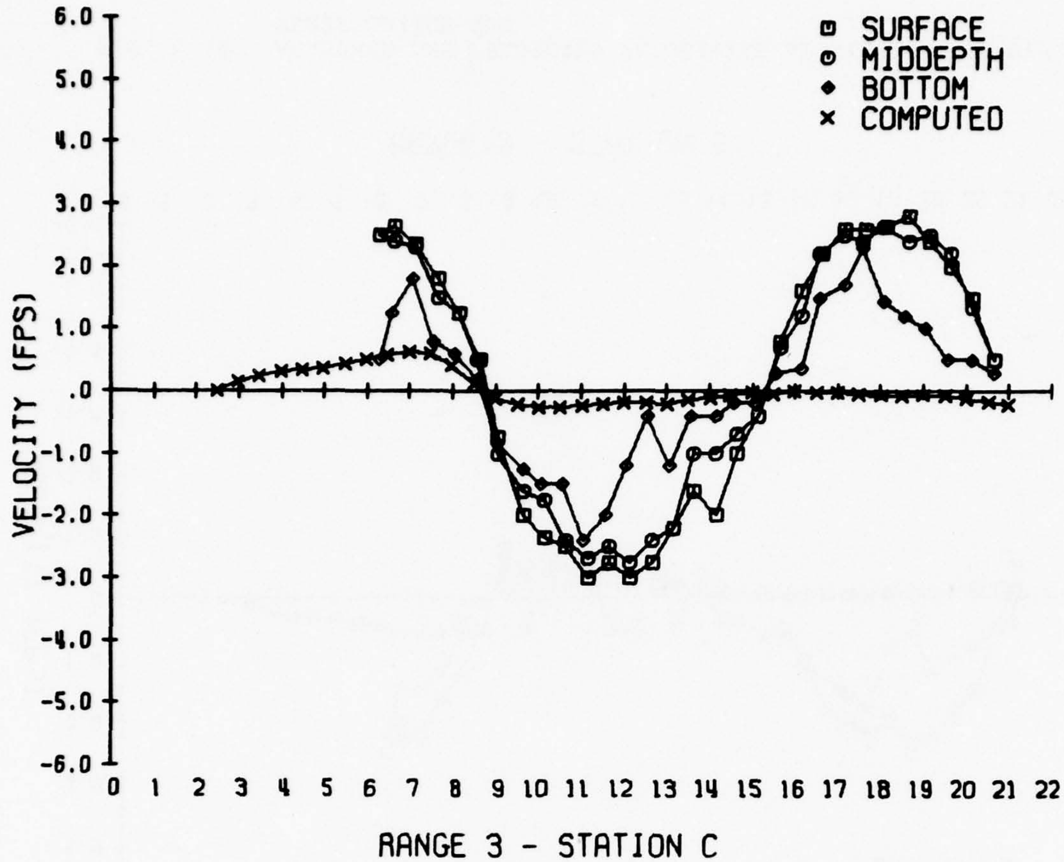


FIG. 4.15 MEASURED AND PREDICTED VELOCITIES AT STATION 3 CENTER, VERIFICATION RUN

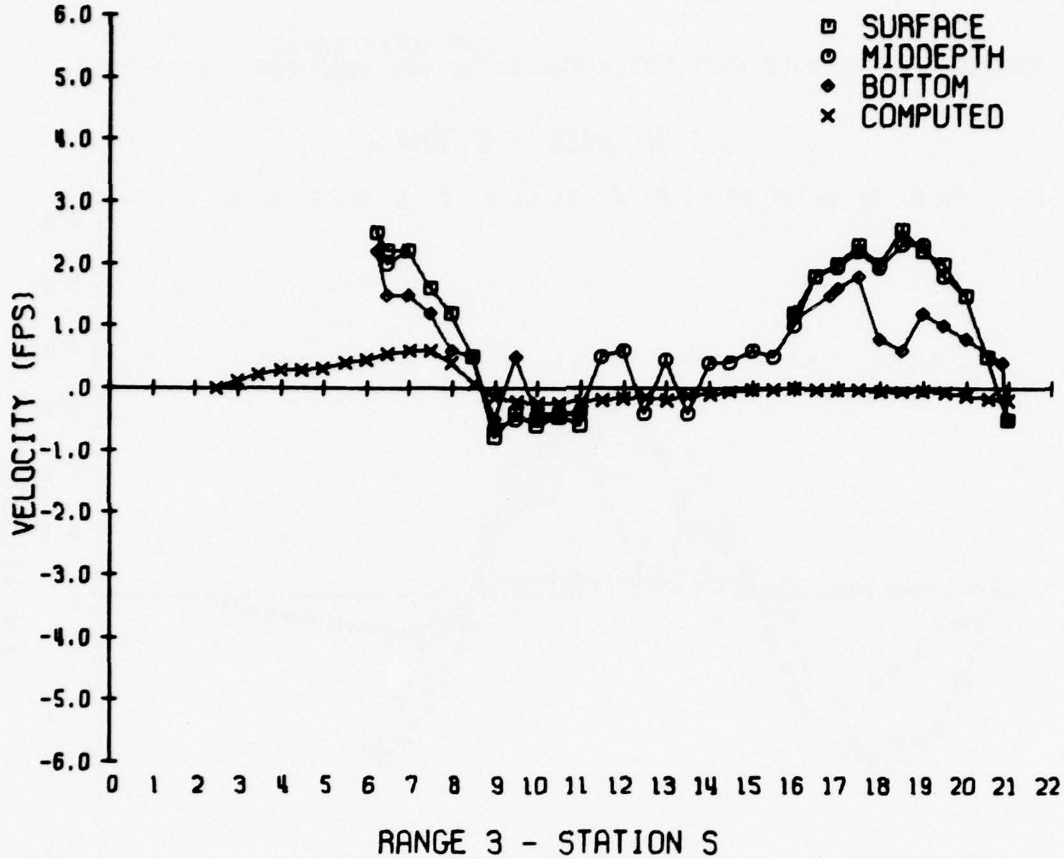


FIG. 4.16 MEASURED AND PREDICTED VELOCITIES AT STATION 3 SOUTH, VERIFICATION RUN

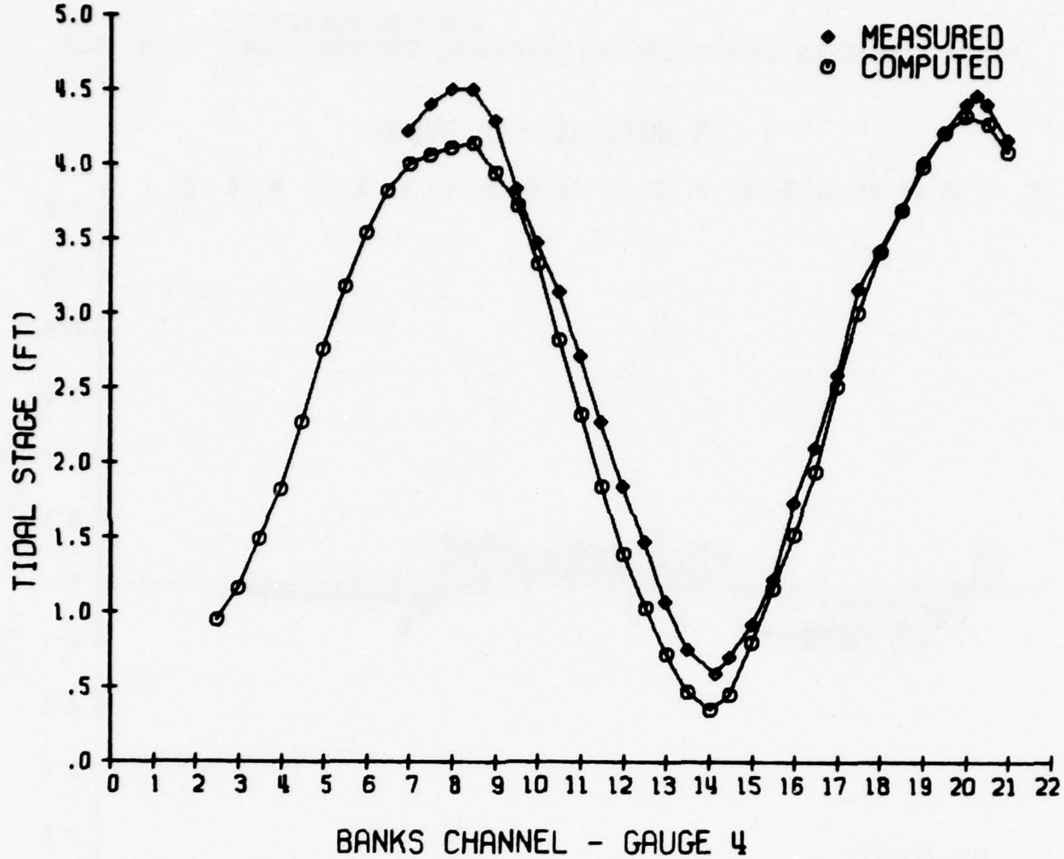


FIG. 4.17 MEASURED AND PREDICTED TIDE FOR STATION 4, VERIFICATION RUN

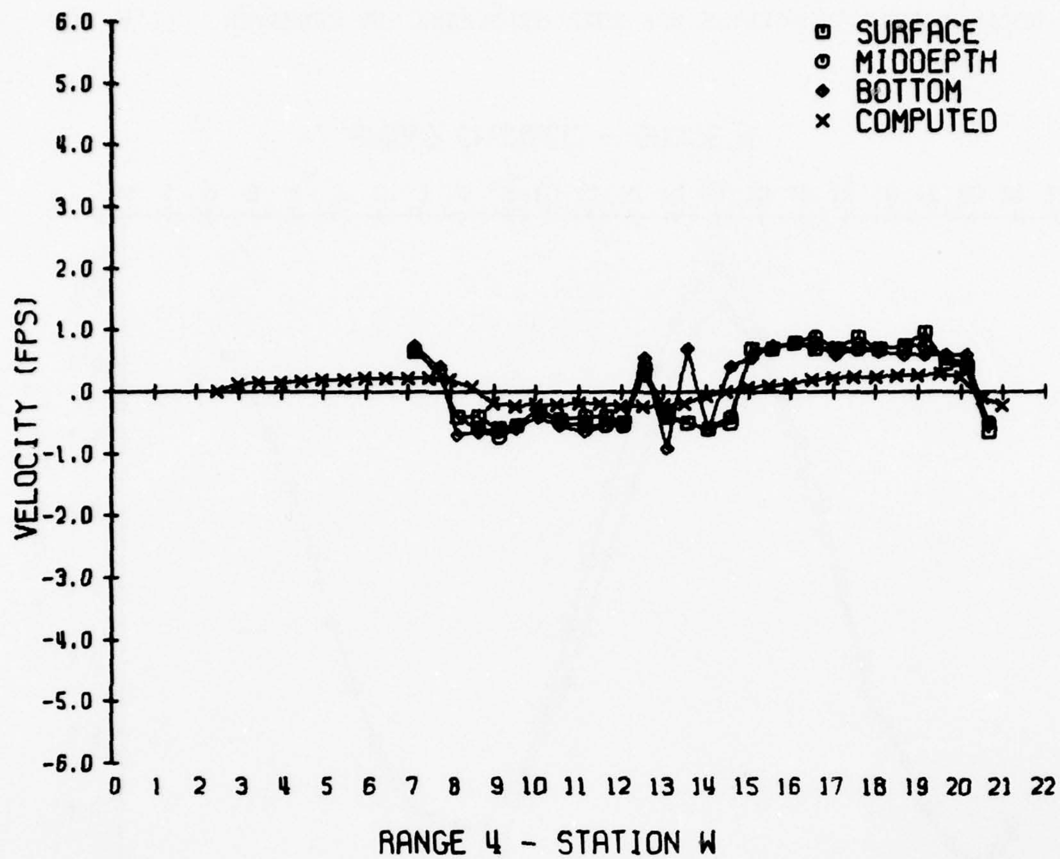


FIG. 4.18 MEASURED AND PREDICTED VELOCITIES AT STATION 4 WEST, VERIFICATION RUN

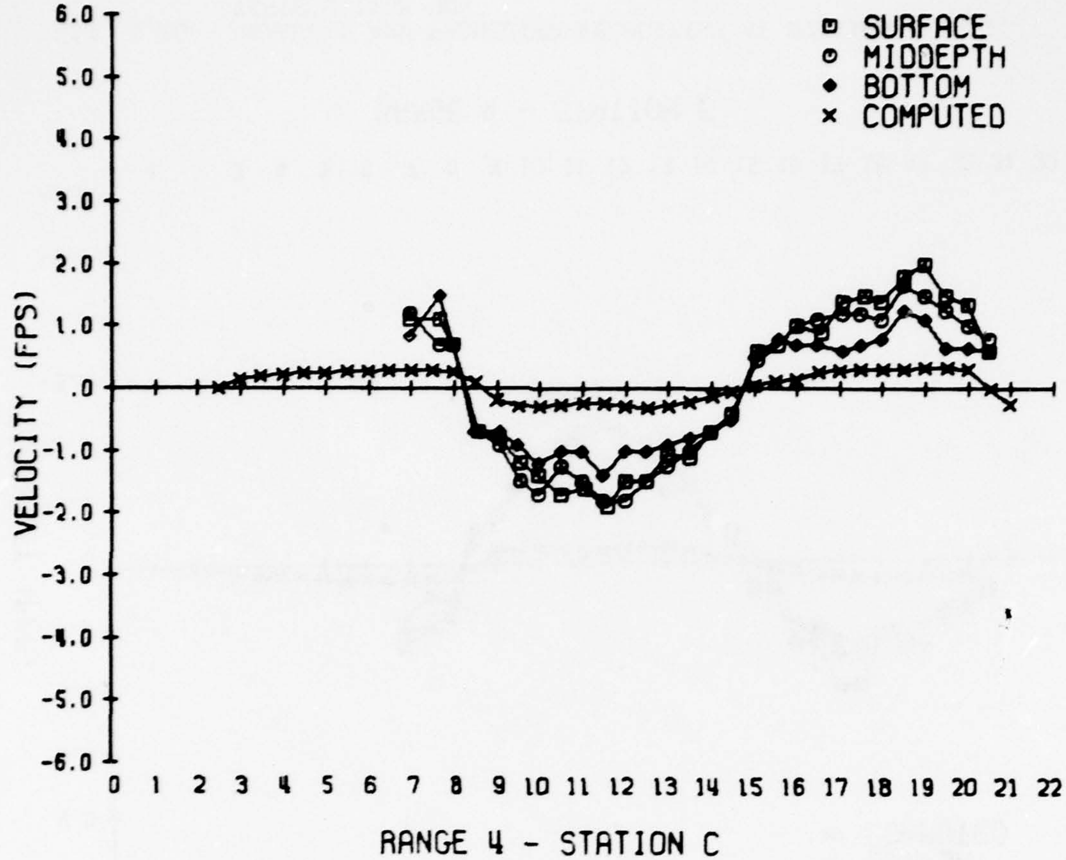


FIG. 4.19 MEASURED AND PREDICTED VELOCITIES AT STATION 4 CENTER, VERIFICATION RUN

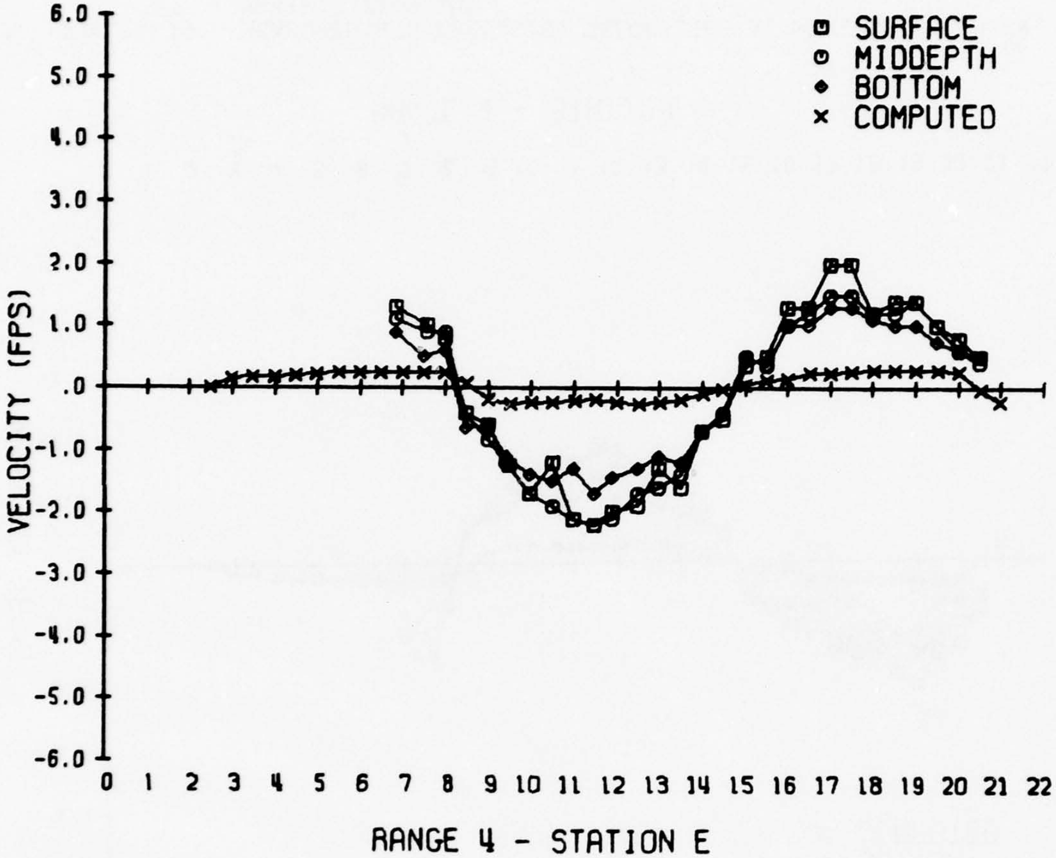


FIG. 4.20 MEASURED AND PREDICTED VELOCITIES AT STATION 4 EAST, VERIFICATION RUN

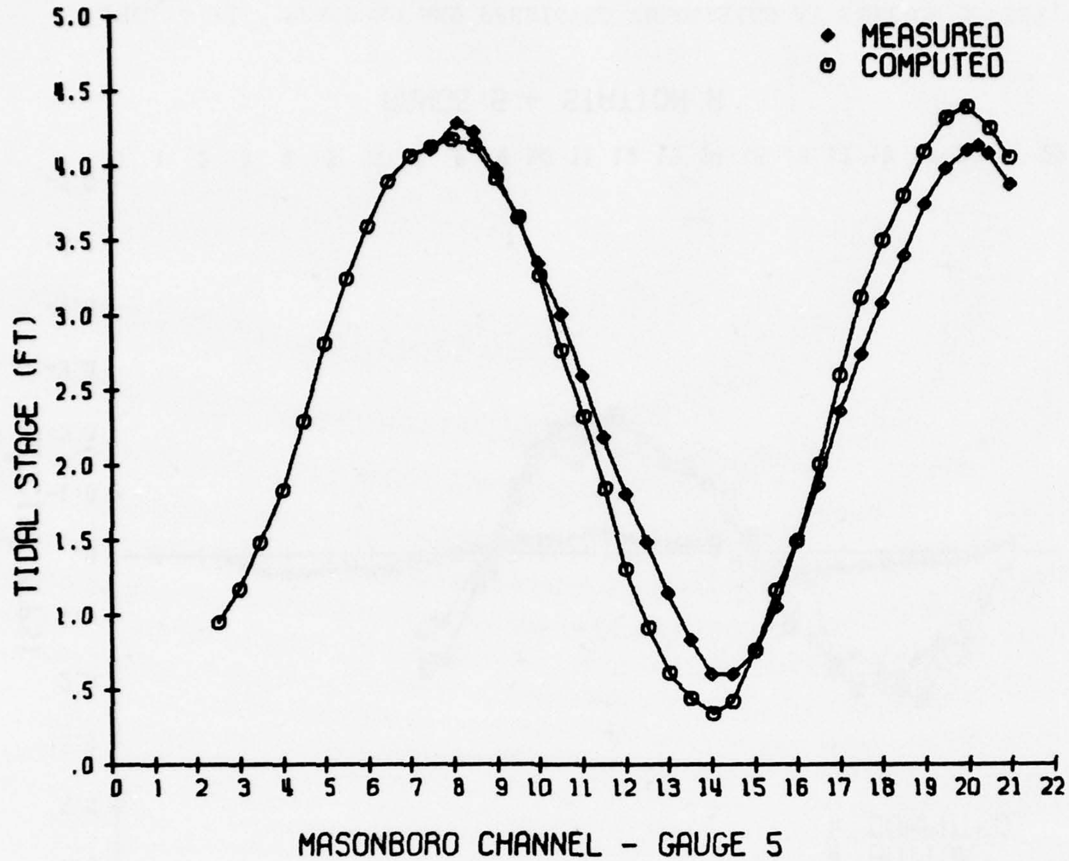


FIG. 4.21 MEASURED AND PREDICTED TIDE FOR STATION 5, VERIFICATION RUN

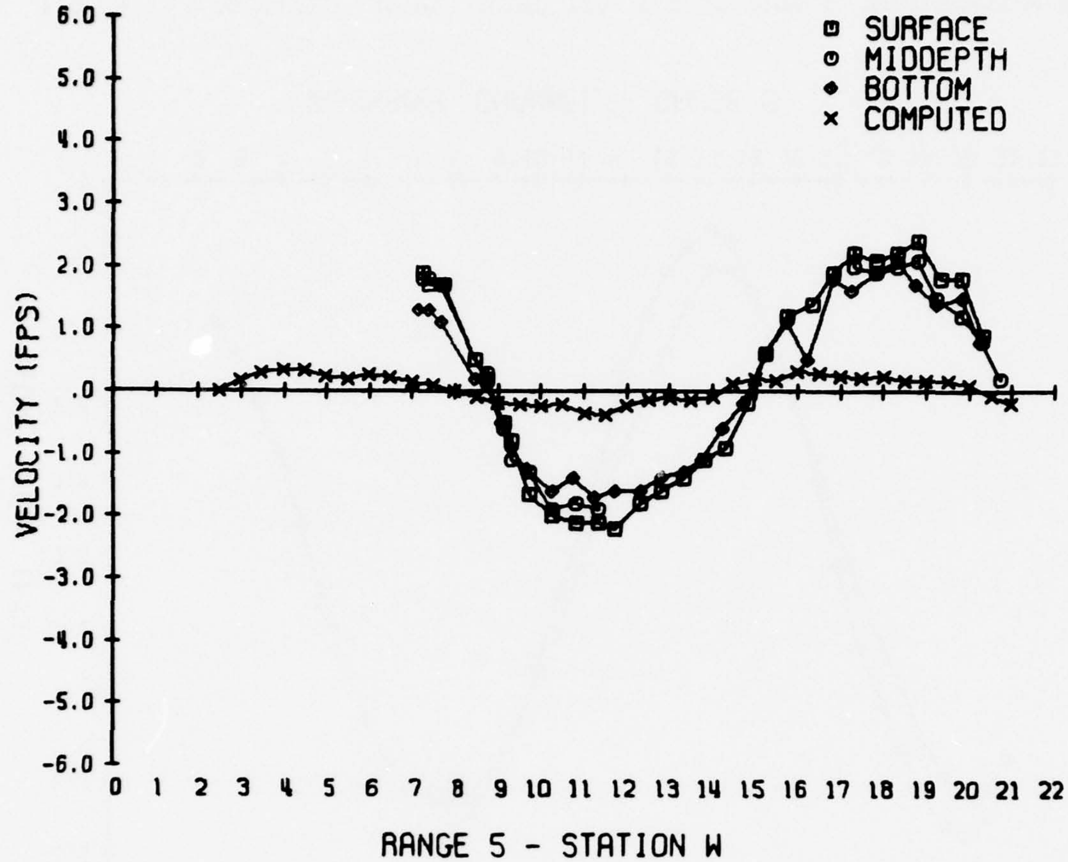


FIG. 4.22 MEASURED AND PREDICTED VELOCITIES AT STATION 5 WEST, VERIFICATION RUN

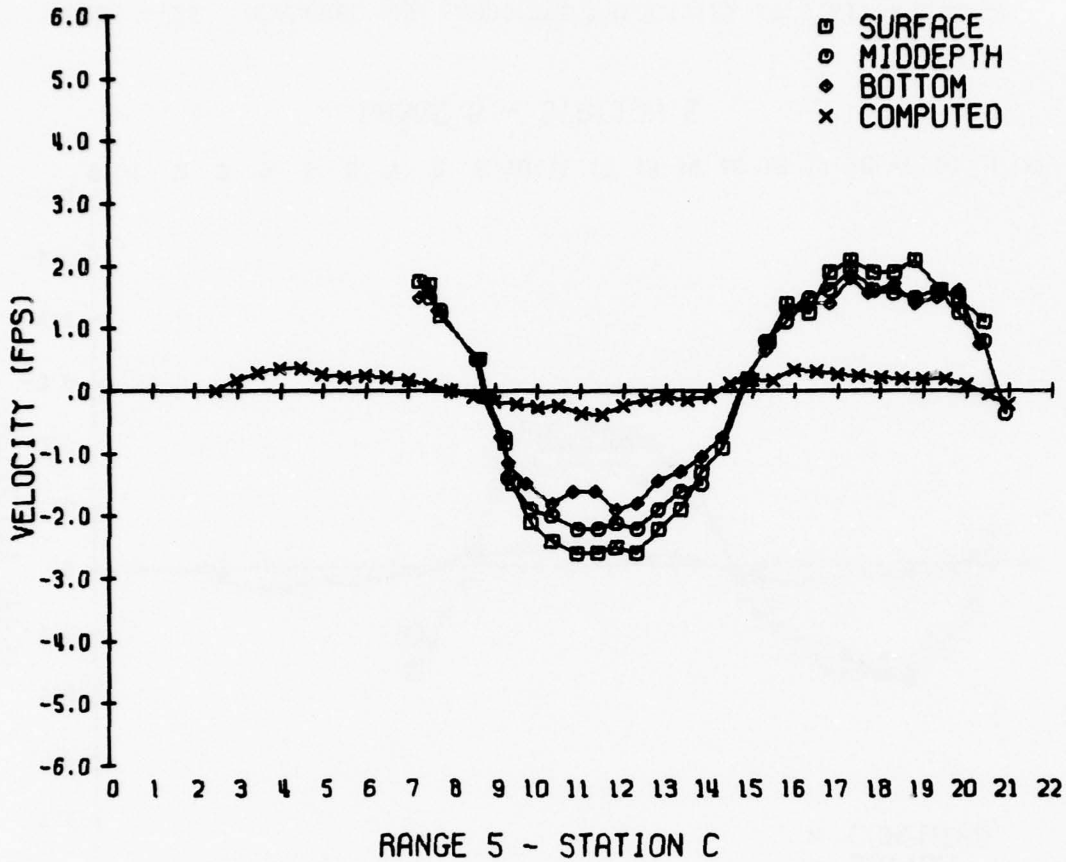


FIG. 4.23 MEASURED AND PREDICTED VELOCITIES AT STATION 5 CENTER, VERIFICATION RUN

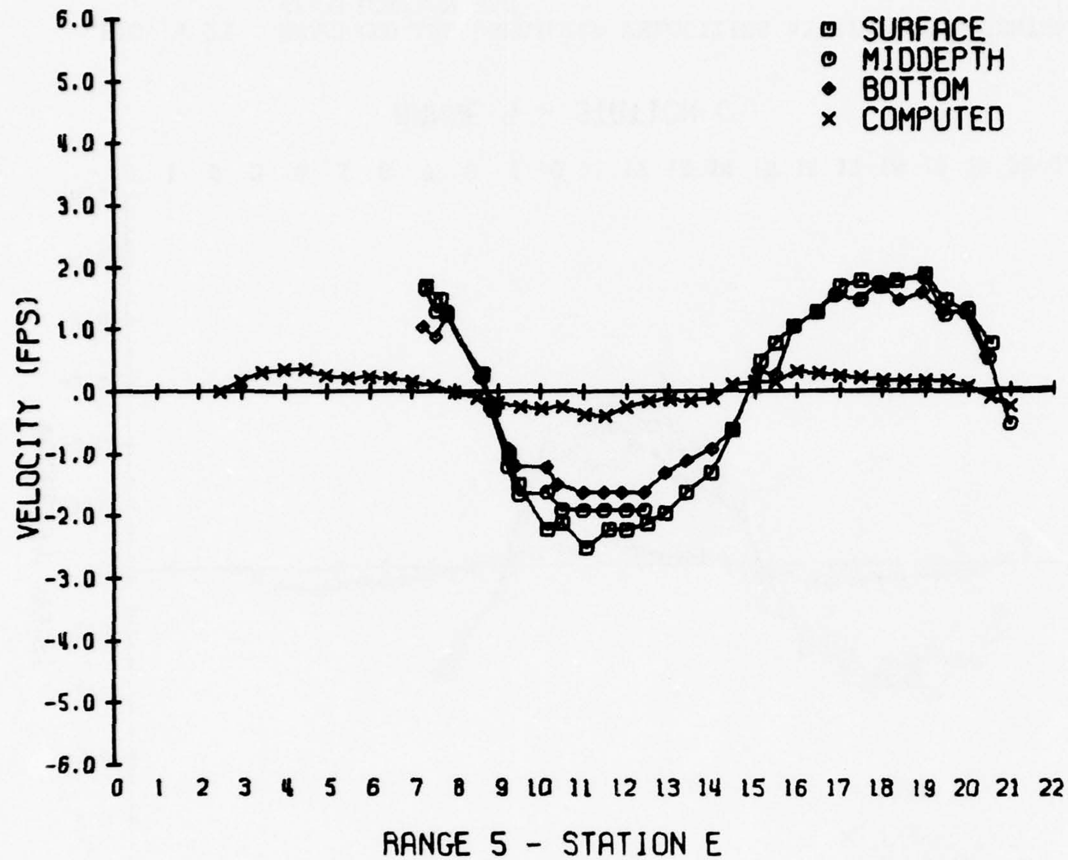


FIG. 4.24 MEASURED AND PREDICTED VELOCITIES AT STATION 5 EAST, VERIFICATION RUN

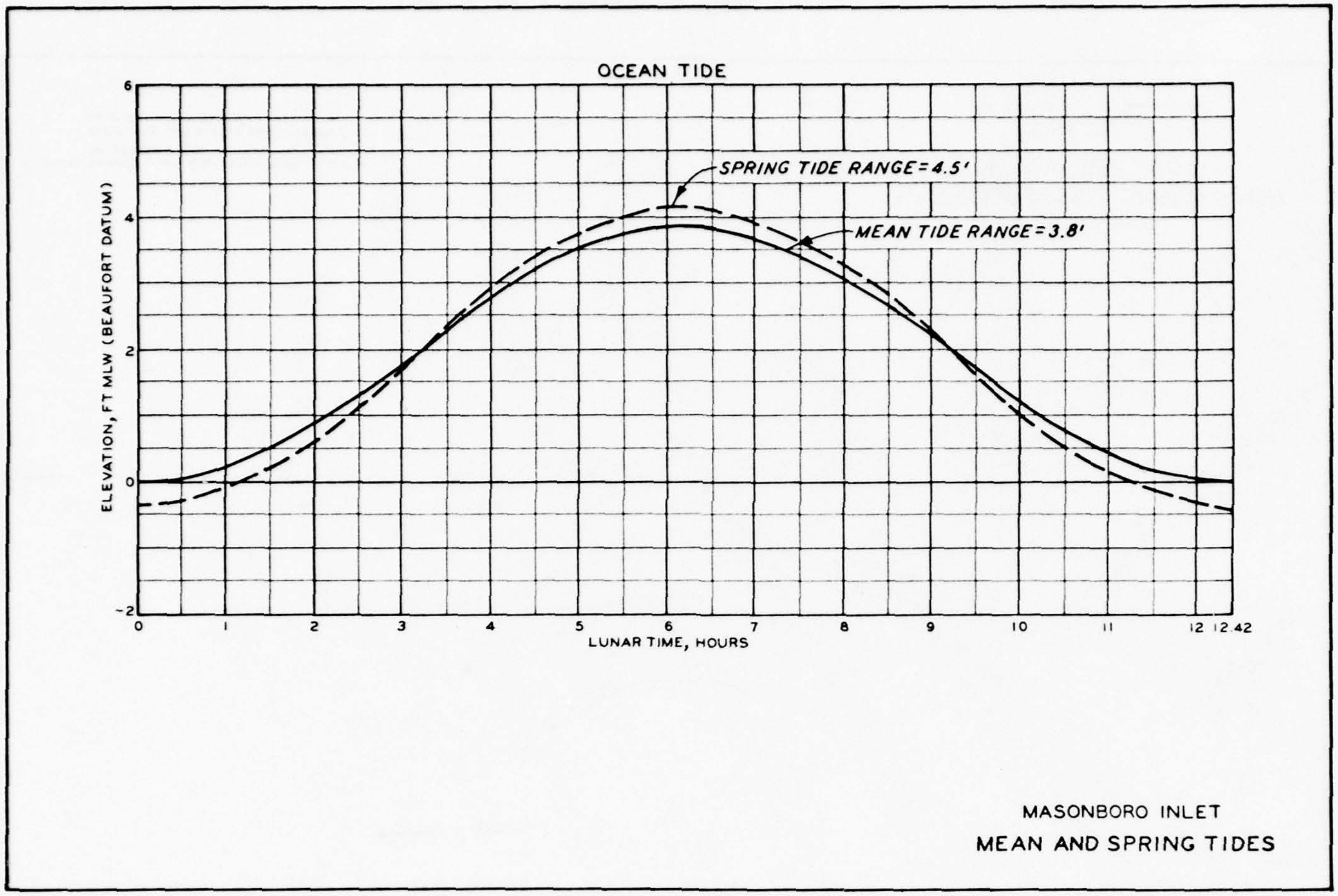


FIG. 4.25 DRIVING TIDES FOR NOVEMBER 1964 AND JUNE 1967 CASES

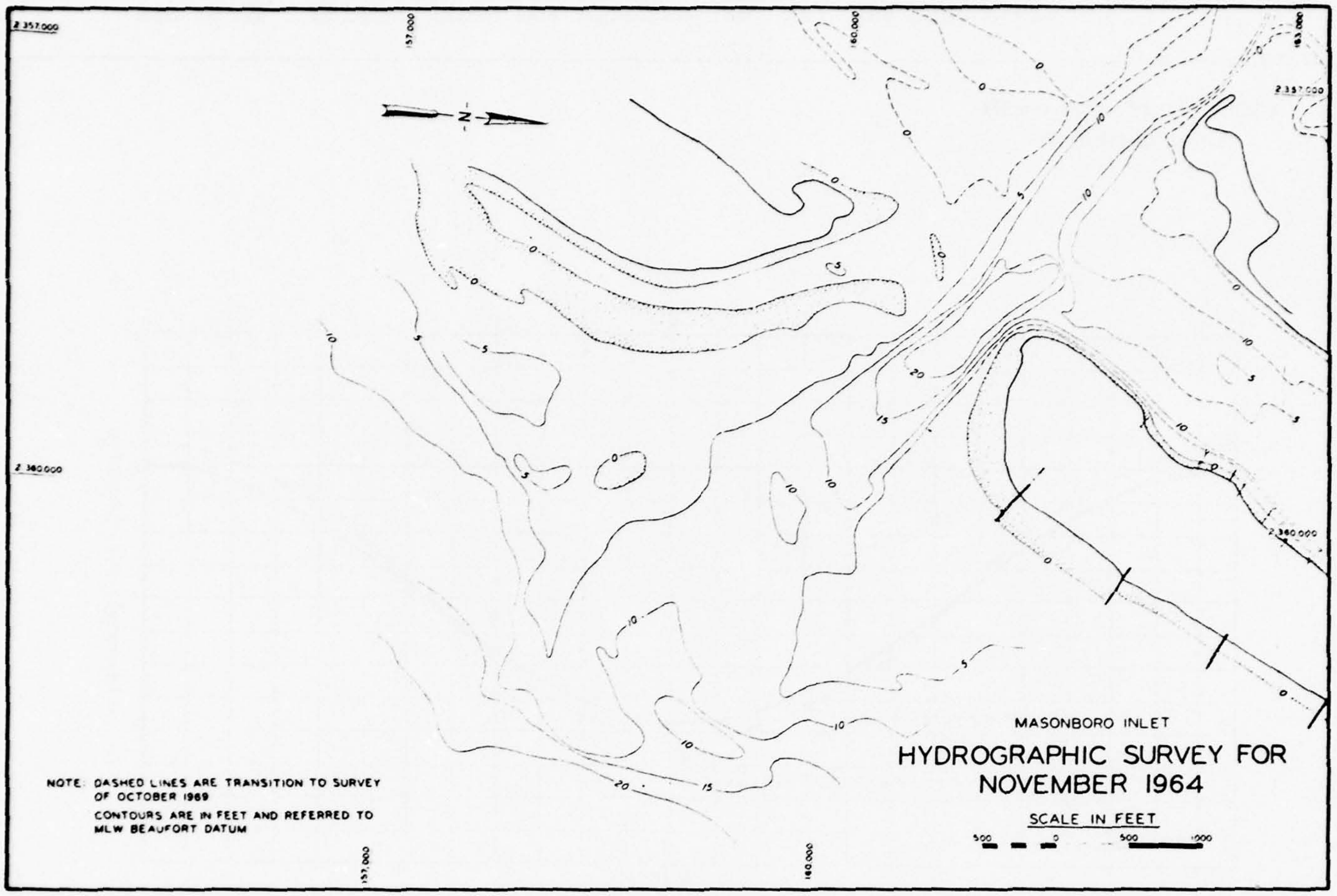


FIG. 4.26 NOVEMBER 1964 SURVEY

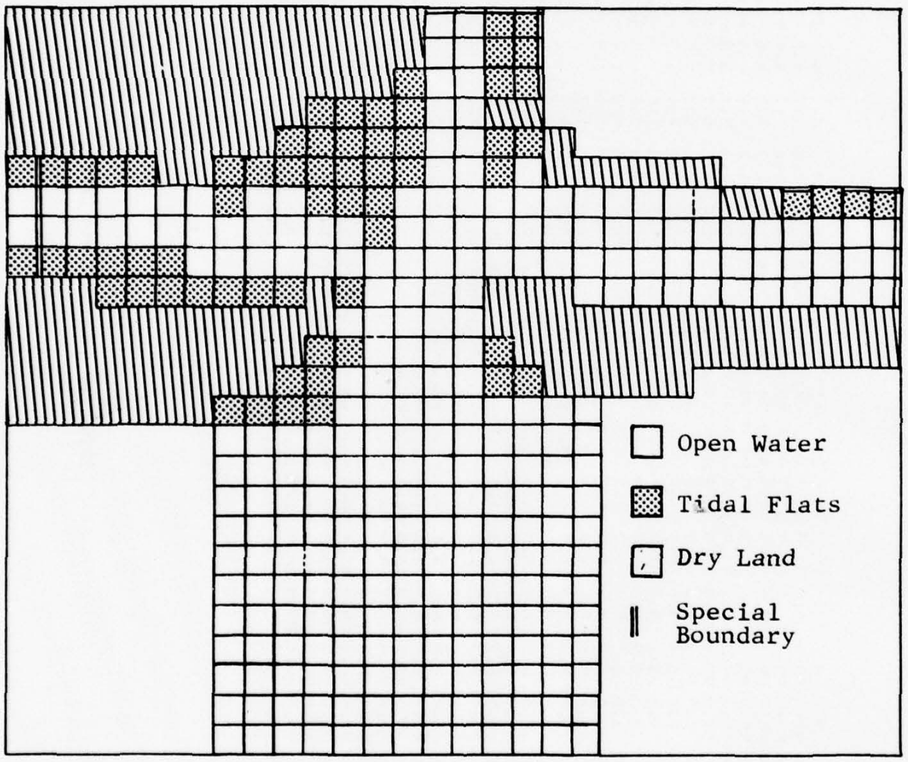


FIG. 4.27 GRID SYSTEM FOR NOVEMBER 1964 CASE

I/J	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-1.8	3.0	4.0	-3.0	99.9	99.9	99.9	99.9	99.9
2	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-1.8	3.3	4.0	-3.0	99.9	99.9	99.9	99.9	99.9
3	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-1.7	3.3	4.6	-2.9	99.9	99.9	99.9	99.9	99.9
4	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-2.3	-1.3	4.6	4.1	-2.8	99.9	99.9	99.9	99.9	99.9
5	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-2.3	-1.0	5.1	3.8	-2.8	99.9	99.9	99.9	99.9	99.9
6	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-2.0	-1.1	6.1	2.2	99.9	99.9	99.9	99.9	99.9	99.9
7	20.0	20.0	17.0	14.0	12.0	9.0	7.0	5.0	5.0	2.0	.7	-.7	99.9	99.9	99.9	-1.5	2.0	6.3	1.5	99.9	99.9	99.9	99.9	99.9	99.9
8	20.0	20.0	17.0	14.0	12.0	9.0	7.0	5.0	5.0	2.0	.7	-.6	99.9	99.9	99.9	-1.5	2.9	6.4	-1.5	-1.2	99.9	99.9	99.9	99.9	99.9
9	19.0	19.0	17.0	13.0	9.0	8.0	5.0	3.0	3.0	2.0	.7	-.5	99.9	99.9	99.9	-1.5	3.3	6.0	.5	-1.0	99.9	99.9	99.9	99.9	99.9
10	18.0	18.0	14.0	11.0	8.0	4.0	2.0	1.0	2.0	2.0	1.0	-.5	-.5	99.9	99.9	-1.5	3.0	5.6	.5	-1.5	-1.0	99.9	99.9	99.9	99.9
11	17.0	17.0	9.0	7.0	6.0	6.0	5.0	2.0	2.0	4.0	2.0	-.5	-.5	-.5	99.9	99.9	4.3	4.3	-1.0	-1.0	-.7	-1.5	99.9	99.9	99.9
12	18.0	18.0	8.0	8.0	9.0	8.0	7.0	6.0	4.0	6.0	5.0	4.0	1.0	-.5	.6	-.5	5.0	2.1	-1.5	-.8	-.6	-2.5	99.9	99.9	99.9
13	20.0	20.0	9.0	10.0	12.0	10.0	8.0	7.0	7.0	8.0	7.0	7.0	4.0	2.0	.6	2.0	5.0	-1.0	-1.5	-.7	-.5	-3.5	99.9	99.9	99.9
14	20.0	20.0	10.0	11.0	11.0	9.0	7.0	7.0	9.0	10.0	10.0	13.0	15.0	14.0	6.0	3.0	5.0	4.0	2.0	-1.0	-.4	-2.0	-3.5	99.9	99.9
15	20.0	20.0	15.0	11.0	10.0	8.0	6.0	6.0	7.0	8.0	11.0	12.0	15.0	20.0	19.0	19.0	16.0	12.0	11.0	12.0	12.0	9.0	6.0	4.0	3.0
16	20.0	20.0	17.0	13.0	10.0	7.0	5.0	5.0	6.0	5.0	5.0	4.0	5.0	6.0	6.0	7.0	15.0	13.0	7.0	8.0	7.0	12.0	13.0	15.0	18.0
17	20.0	20.0	19.0	15.0	14.0	9.0	6.0	5.0	4.0	4.0	3.0	2.0	-.5	-1.0	99.9	99.9	10.0	5.0	5.0	-1.0	-1.0	99.9	-1.4	-1.5	-1.6
18	22.0	22.0	20.0	16.0	15.0	13.0	9.0	6.0	5.0	4.0	3.0	1.0	-1.0	99.9	99.9	99.9	6.0	6.0	5.0	1.3	-2.0	99.9	-1.9	-1.9	-2.0
19	22.0	22.0	22.0	20.0	15.0	15.0	11.0	8.0	6.0	4.0	3.0	1.0	99.9	99.9	99.9	99.9	8.0	8.0	6.0	99.9	99.9	99.9	99.9	99.9	99.9
20	22.0	22.0	22.0	20.0	15.0	15.0	11.0	8.0	6.0	4.0	3.0	1.0	99.9	99.9	99.9	4.0	7.0	7.0	5.0	99.9	99.9	99.9	99.9	99.9	99.9
21	22.0	22.0	20.0	20.0	15.0	15.0	11.0	8.0	6.0	4.0	3.0	1.0	99.9	99.9	99.9	1.7	11.6	3.8	5.0	99.9	99.9	99.9	99.9	99.9	99.9
22	22.0	22.0	20.0	20.0	15.0	15.0	11.0	8.0	6.0	4.0	3.0	1.0	99.9	99.9	99.9	1.4	11.8	8.7	4.0	99.9	99.9	99.9	99.9	99.9	99.9
23	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	3.5	12.3	12.4	3.0	99.9	99.9	99.9	99.9	99.9	99.9
24	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	5.6	13.2	11.9	3.0	99.9	99.9	99.9	99.9	99.9	99.9
25	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	6.4	15.8	10.8	99.9	99.9	99.9	99.9	99.9	99.9	99.9
26	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	5.6	17.3	6.0	99.9	99.9	99.9	99.9	99.9	99.9	99.9
27	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	6.3	17.6	6.0	-.5	99.9	99.9	99.9	99.9	99.9	99.9
28	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	8.0	17.0	8.0	-.5	99.9	99.9	99.9	99.9	99.9	99.9
29	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	8.0	13.5	1.5	-.5	99.9	99.9	99.9	99.9	99.9	99.9
30	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	8.0	13.0	2.0	-.5	99.9	99.9	99.9	99.9	99.9	99.9

FIG. 4.28 DEPTHS FOR NOVEMBER 1964 CASE

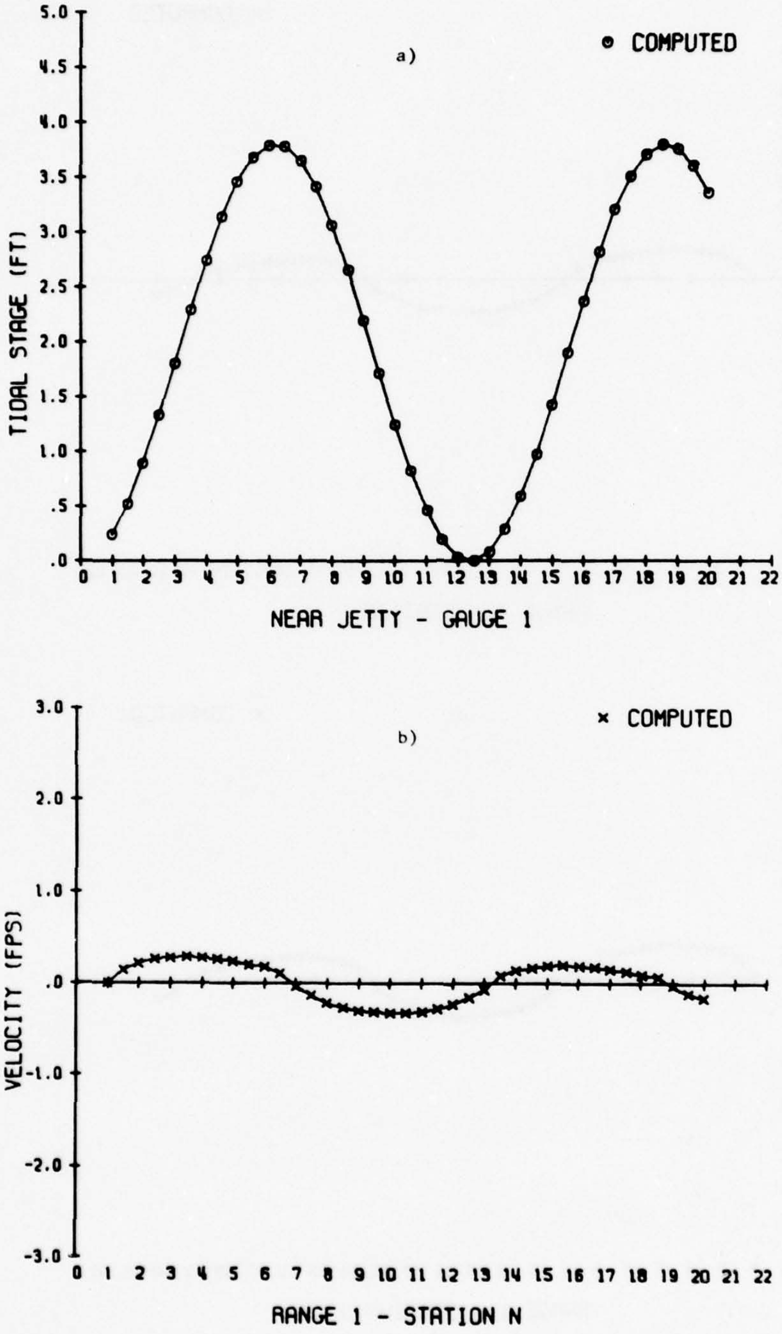


FIG. 4.29 RESULTS OF RUN FOR NOVEMBER 1964, MEAN TIDE FOR STATION 1, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH

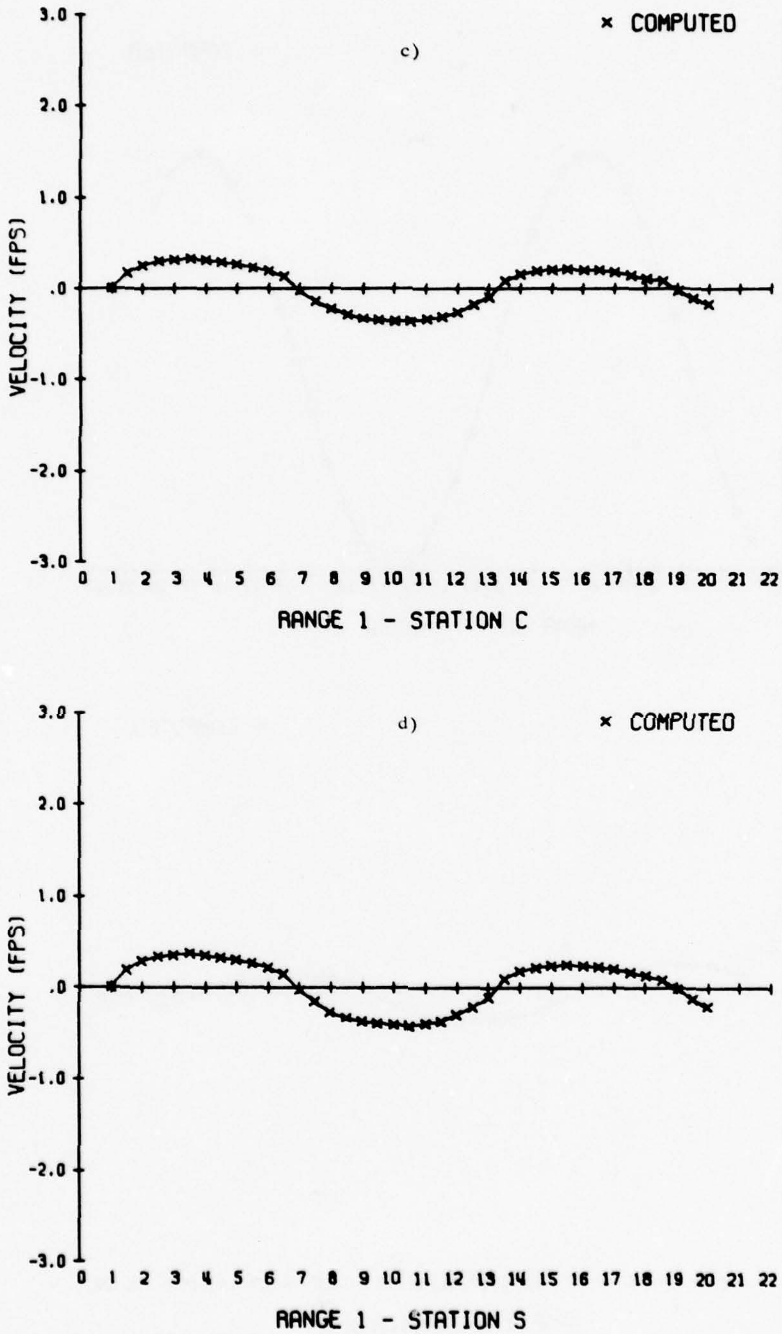


FIG. 4.29 (Continued) RESULTS OF RUN FOR NOVEMBER 1964, MEAN TIDE FOR STATION 1, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH

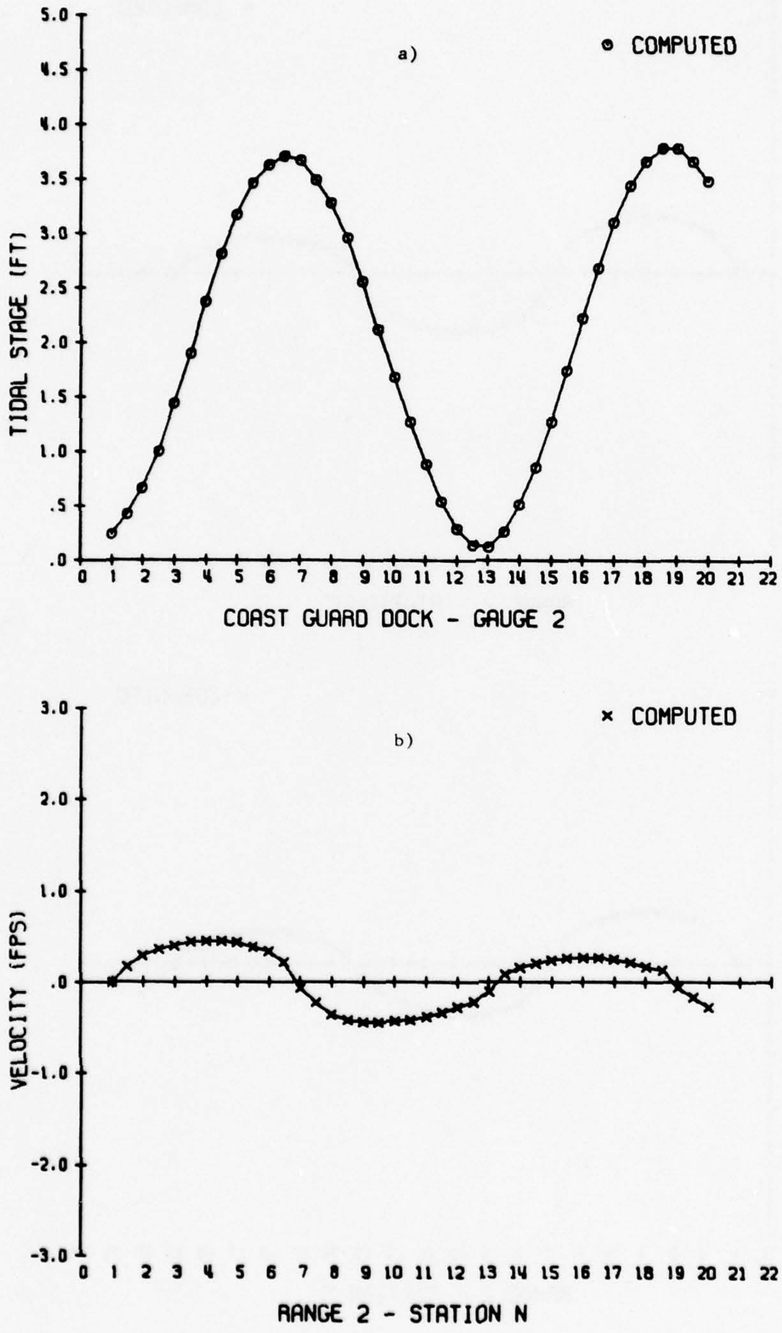
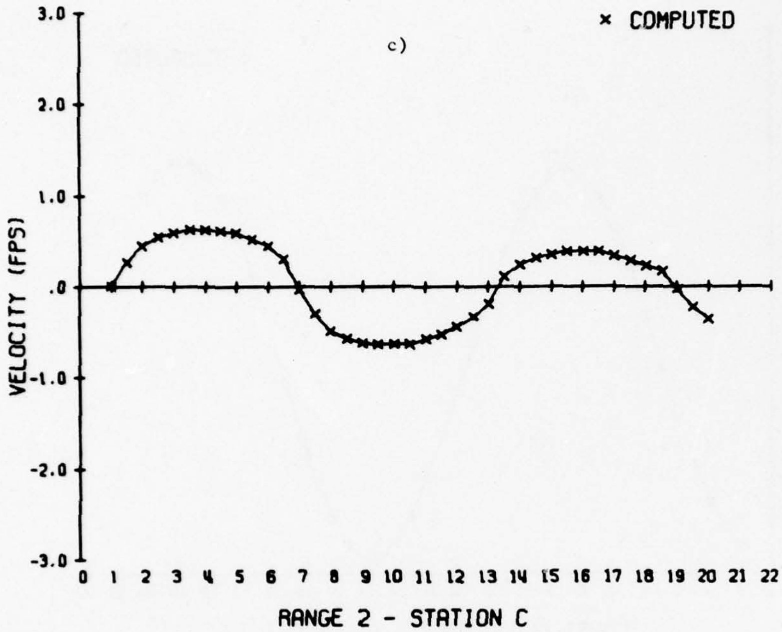


FIG. 4.30 RESULTS OF RUN FOR NOVEMBER 1964, MEAN TIDE FOR STATION 2, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



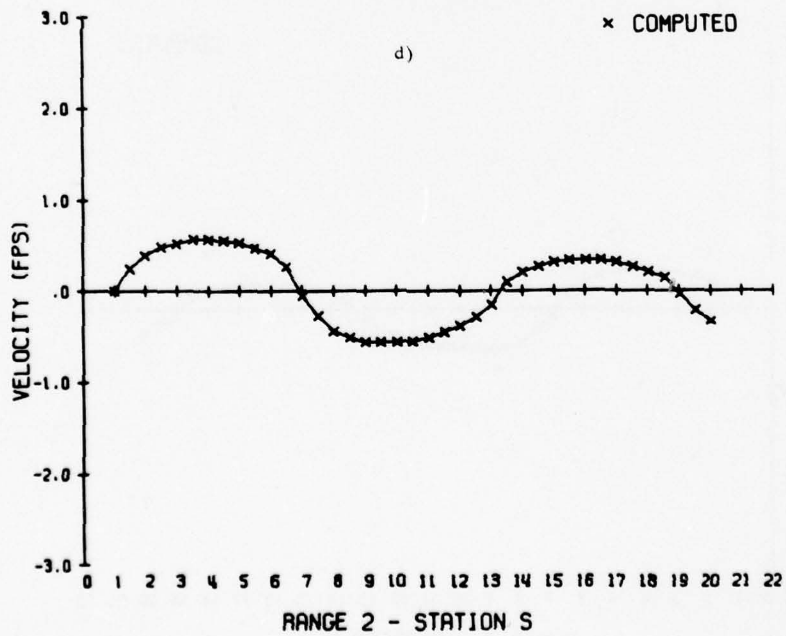


FIG. 4.30 (Continued) RESULTS OF RUN FOR NOVEMBER 1964, MEAN TIDE FOR STATION 2, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH

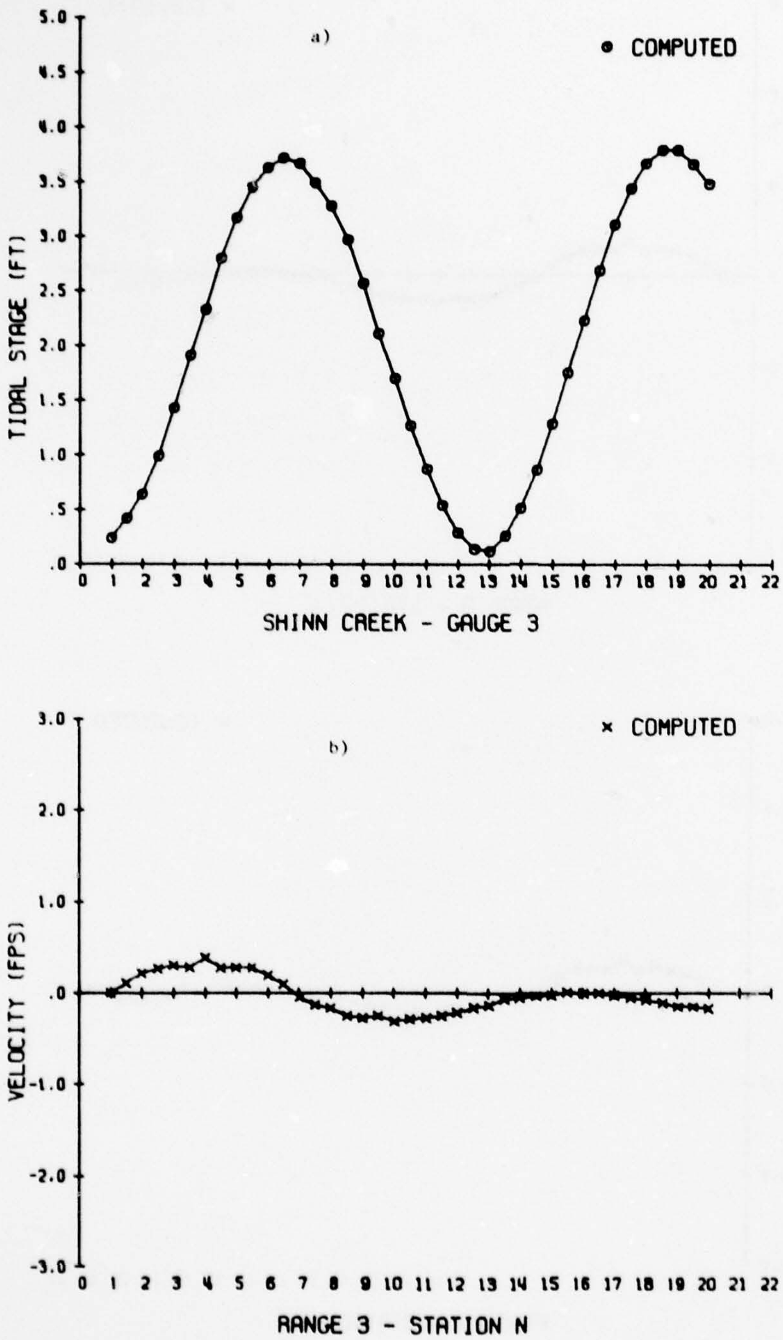
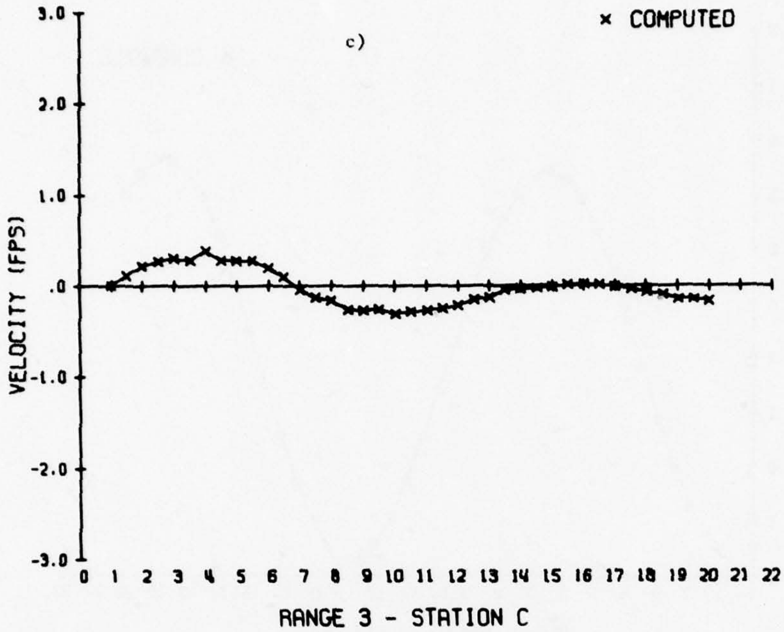


FIG. 4.31 RESULTS OF RUN FOR NOVEMBER 1964, MEAN TIDE FOR STATION 3, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



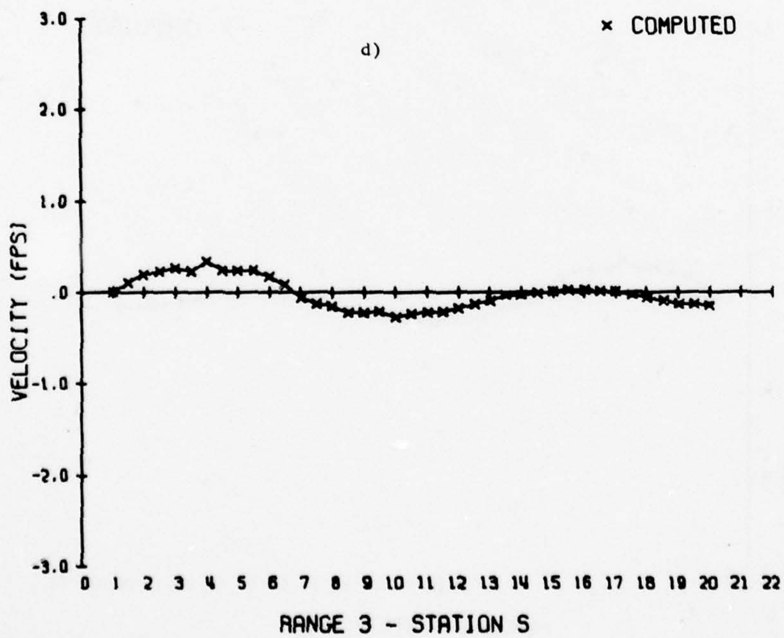


FIG. 4.31 (Continued) RESULTS OF RUN FOR NOVEMBER 1964, MEAN TIDE FOR STATION 3, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH

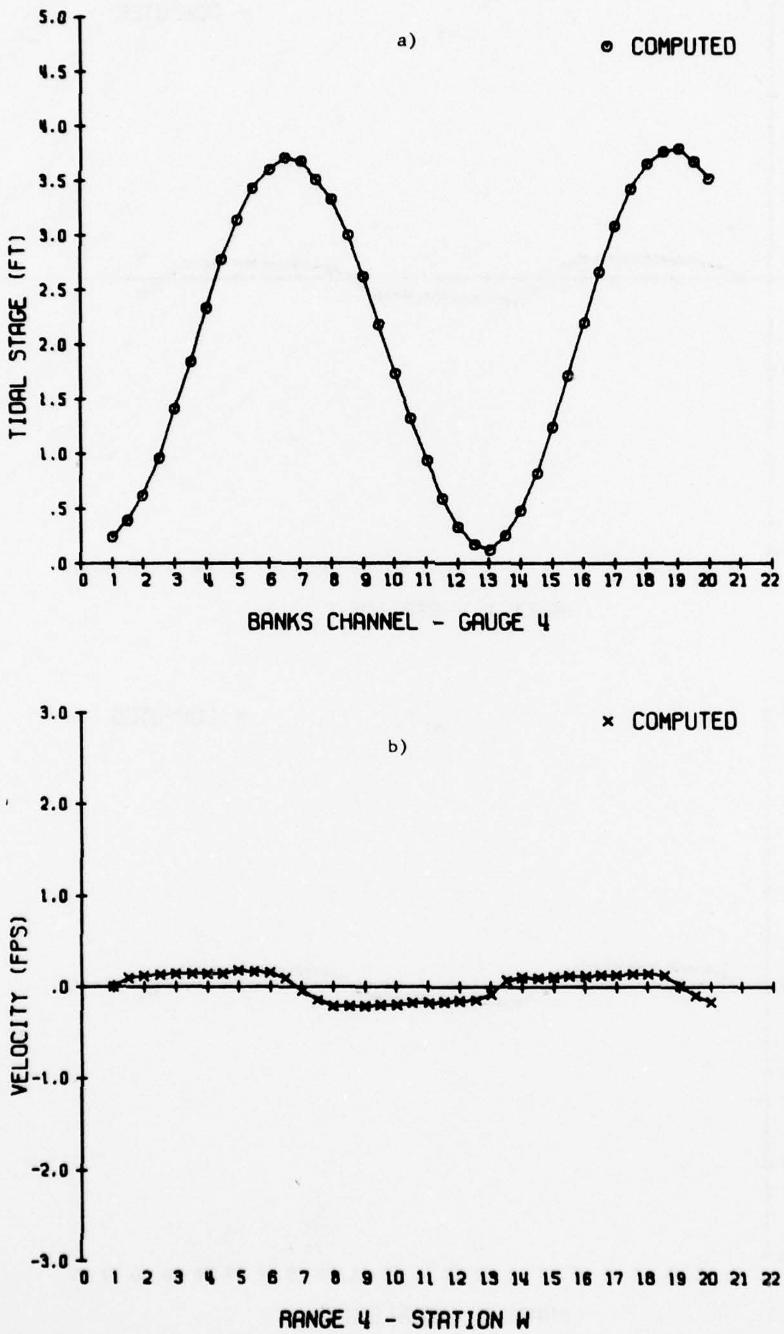
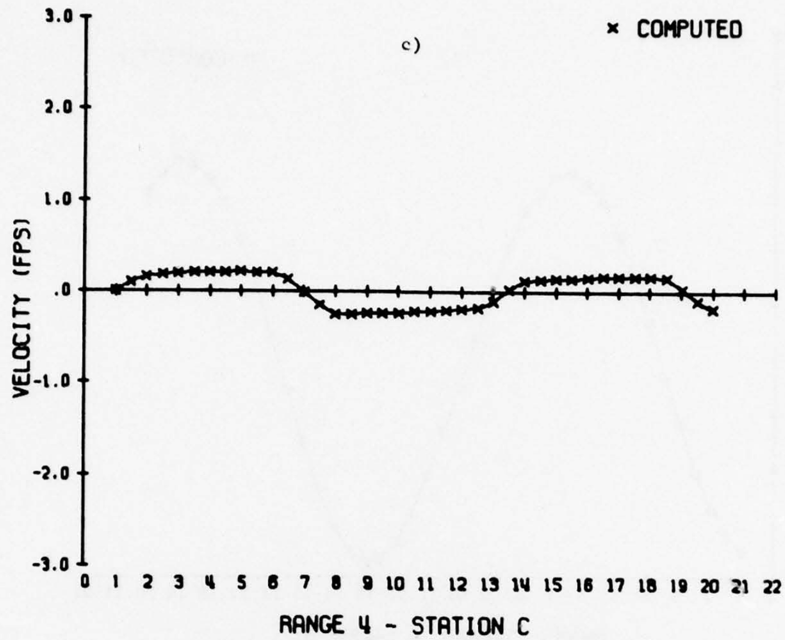


FIG. 4.32 RESULTS OF RUN FOR NOVEMBER 1964, MEAN TIDE FOR STATION 4, a) TIDE, b) VELOCITY WEST, c) VELOCITY CENTER, d) VELOCITY EAST



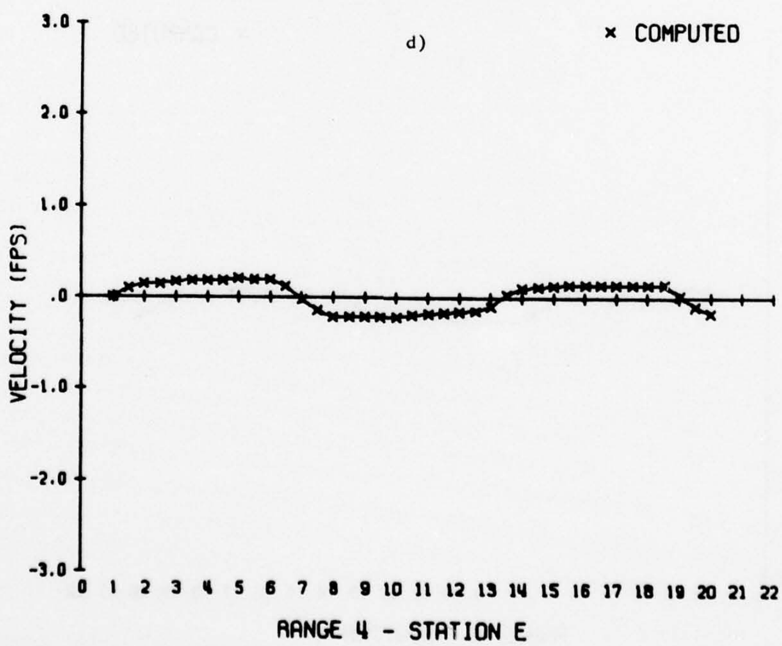
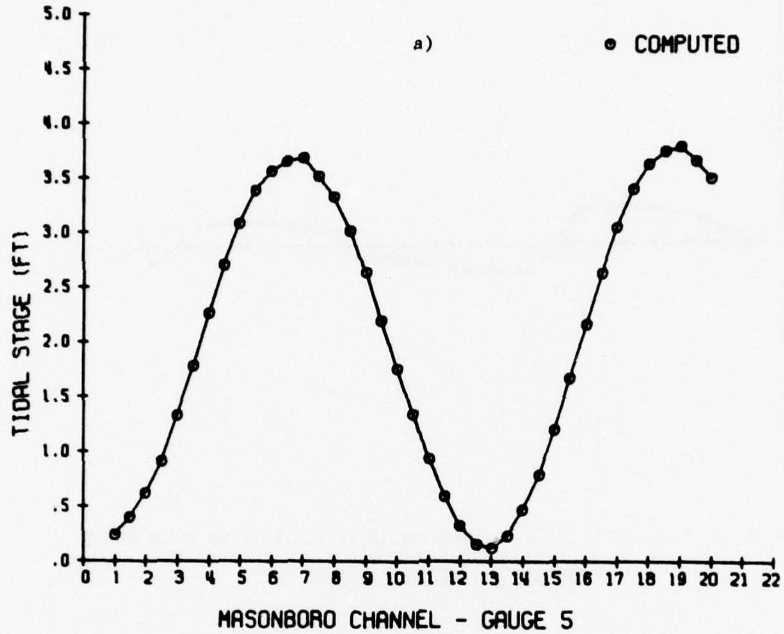


FIG. 4.32 (Continued) RESULTS OF RUN FOR NOVEMBER 1964, MEAN TIDE FOR STATION 4, a) TIDE, b) VELOCITY WEST, c) VELOCITY CENTER, d) VELOCITY EAST



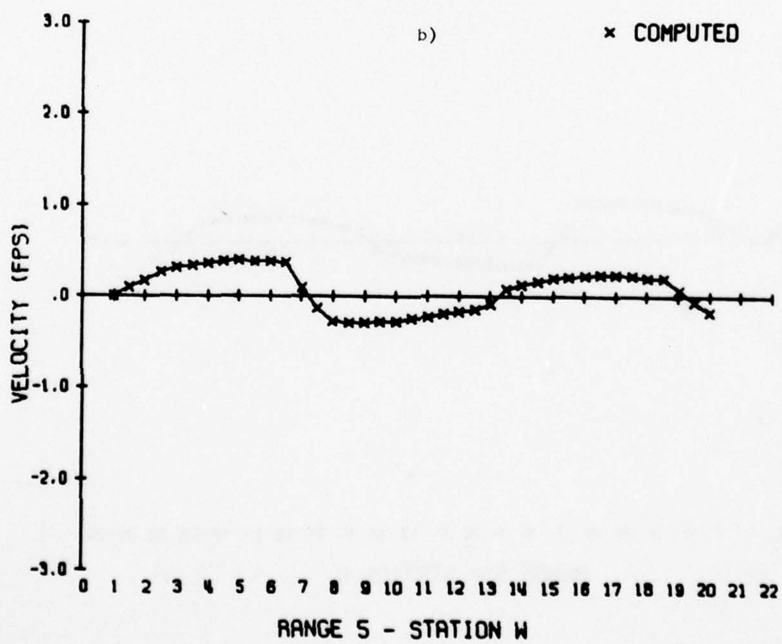
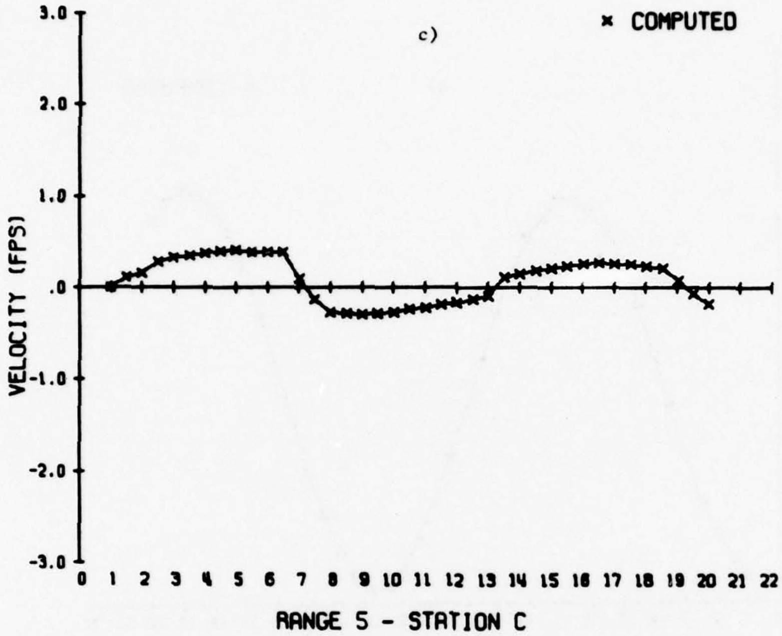


FIG. 4.33 RESULTS OF RUN FOR NOVEMBER 1964, MEAN TIDE FOR STATION 5, a) TIDE, b) VELOCITY WEST, c) VELOCITY CENTER, d) VELOCITY EAST



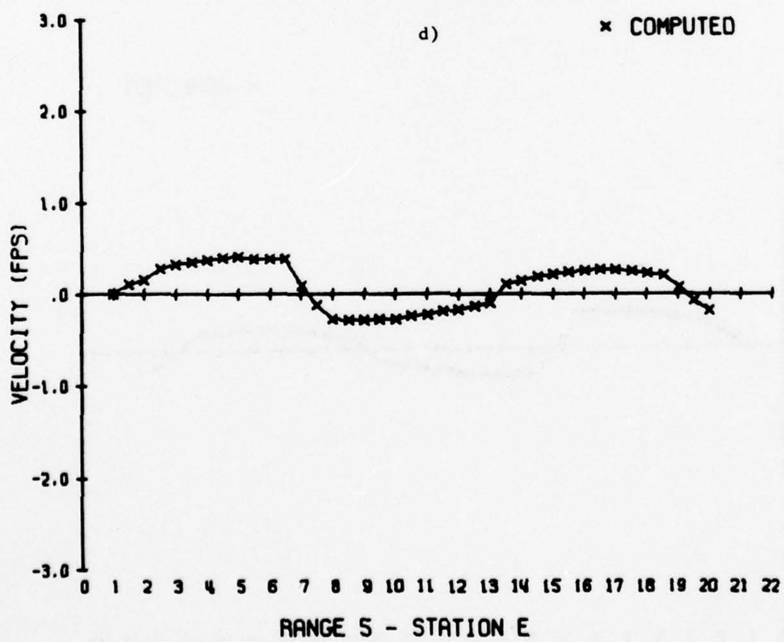
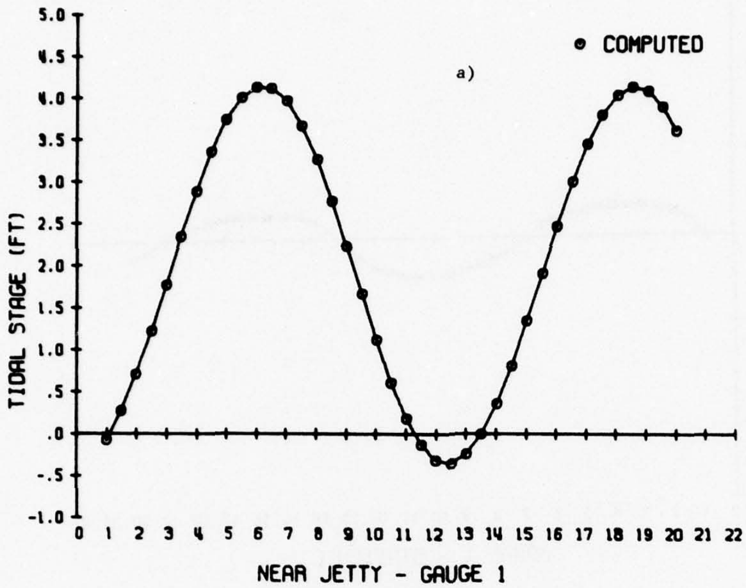


FIG. 4.33 (Continued) RESULTS OF RUN FOR NOVEMBER 1964, MEAN TIDE FOR STATION 5, a) TIDE, b) VELOCITY WEST, c) VELOCITY CENTER, d) VELOCITY EAST



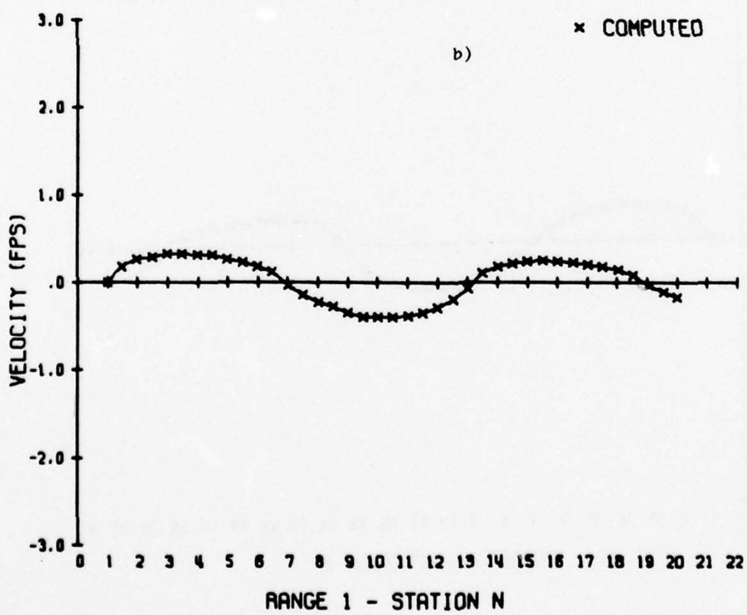
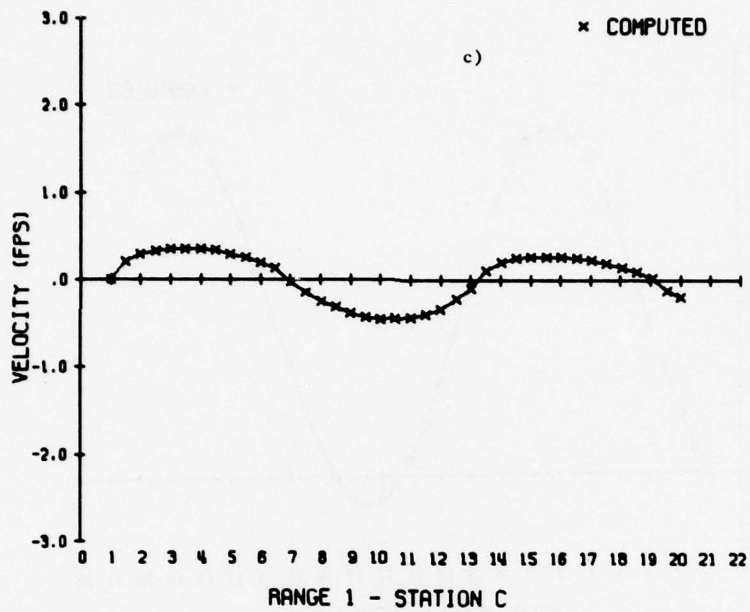


FIG. 4.34 RESULTS OF RUN FOR NOVEMBER 1964, SPRING TIDE FOR STATION 1, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



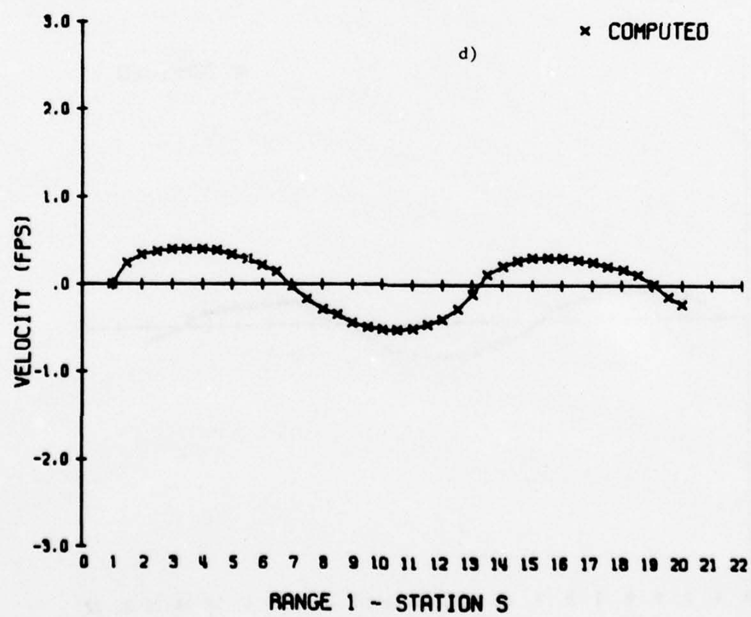
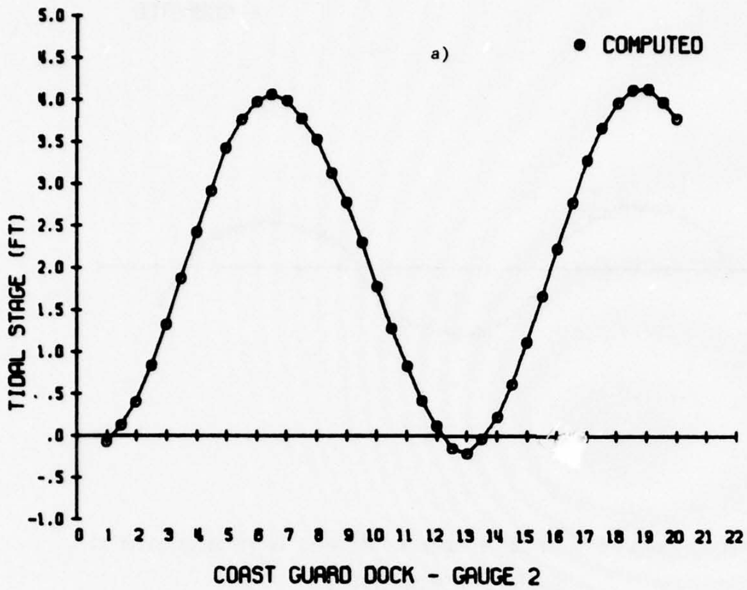


FIG. 4.34 (Continued) RESULTS OF RUN FOR NOVEMBER 1964, SPRING TIDE FOR STATION 1, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



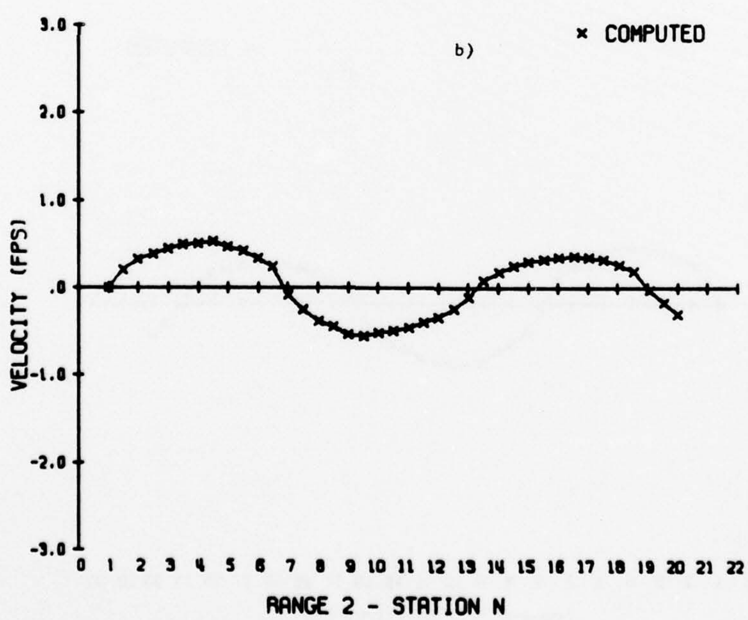
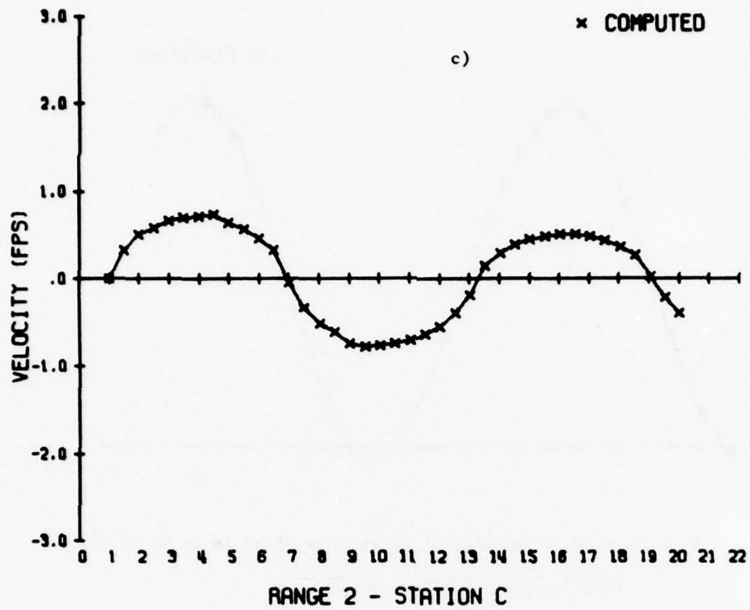


FIG. 4.35 RESULTS OF RUN FOR NOVEMBER, 1964, SPRING TIDE FOR STATION 2, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



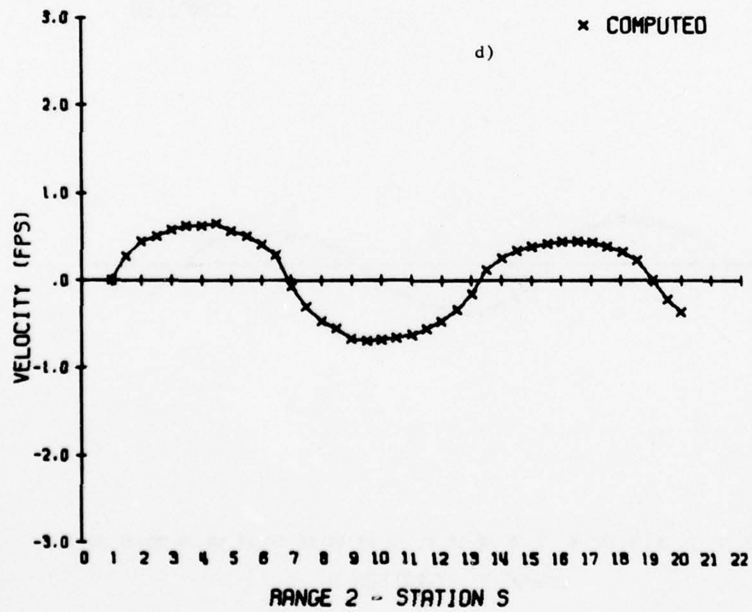
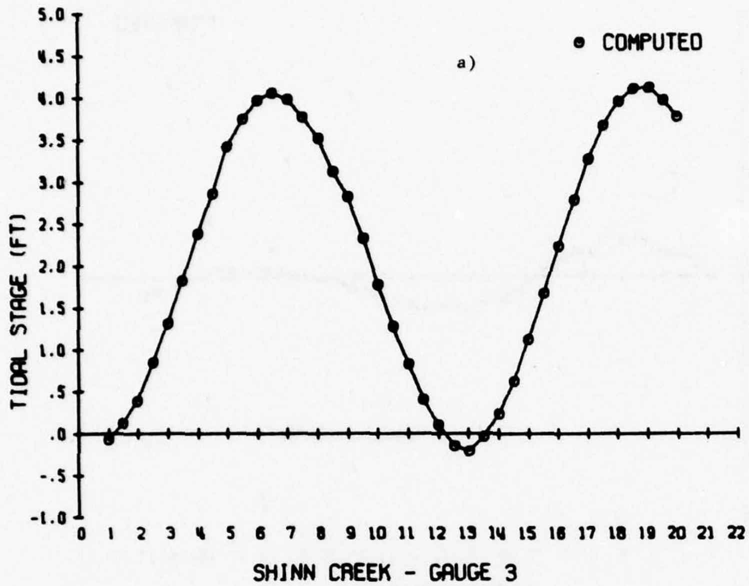


FIG. 4.35 (Continued) RESULTS OF RUN FOR NOVEMBER, 1964, SPRING TIDE FOR STATION 2, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



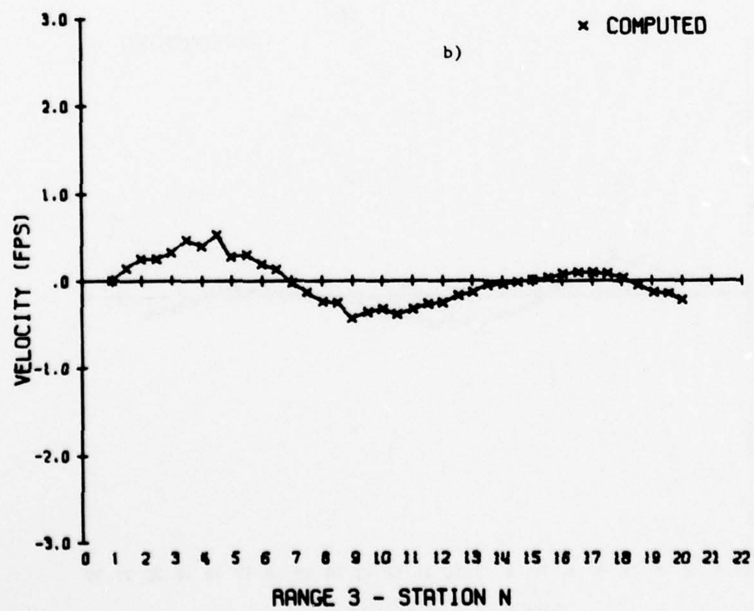
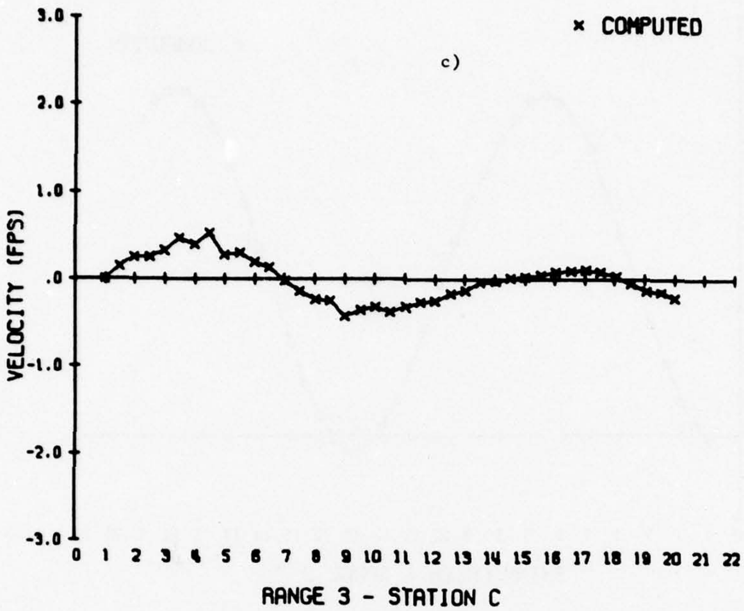


FIG. 4.36 RESULTS OF RUN FOR NOVEMBER, 1964, SPRING TIDE FOR STATION 3, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



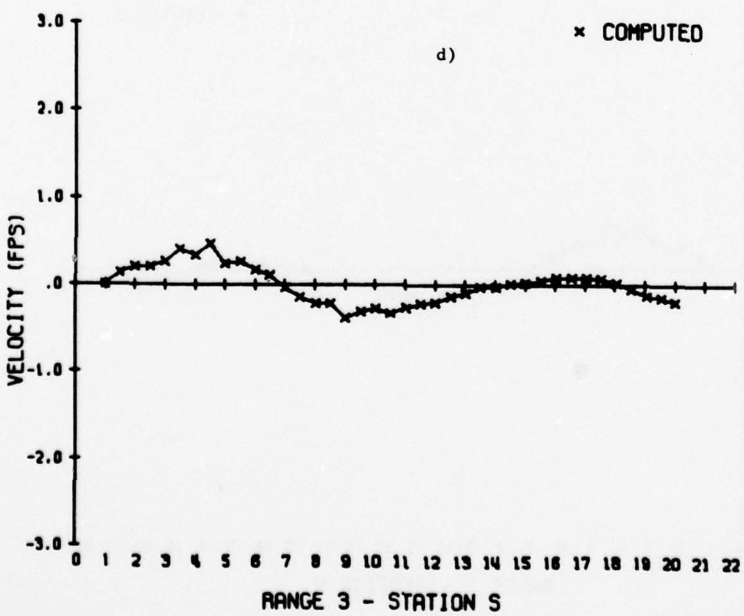
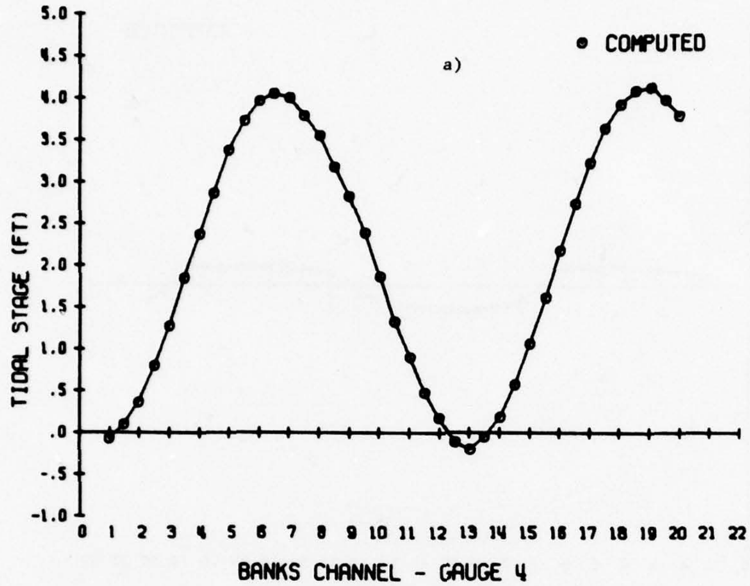


FIG. 4.36 (Continued) RESULTS OF RUN FOR NOVEMBER, 1964, SPRING TIDE FOR STATION 3, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



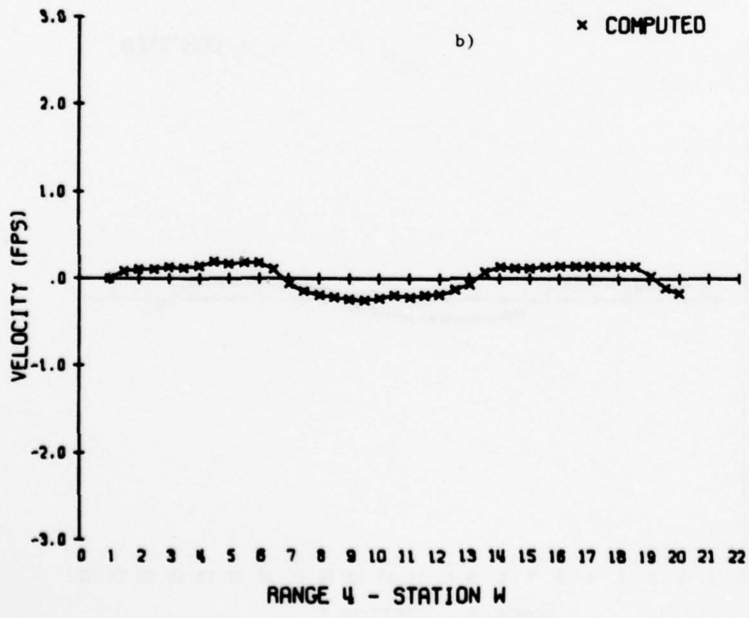
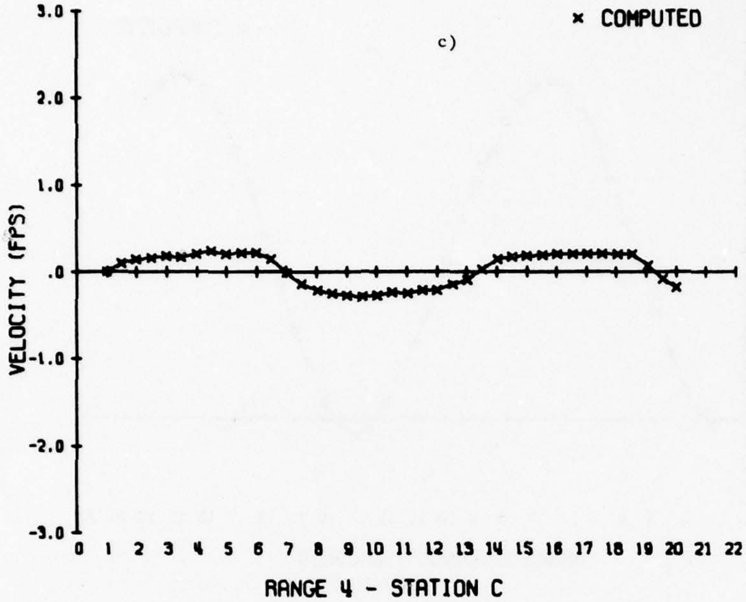


FIG. 4.37 RESULTS OF RUN FOR NOVEMBER, 1964, SPRING TIDE FOR STATION 4, a) TIDE, b) VELOCITY WEST, c) VELOCITY CENTER, d) VELOCITY EAST



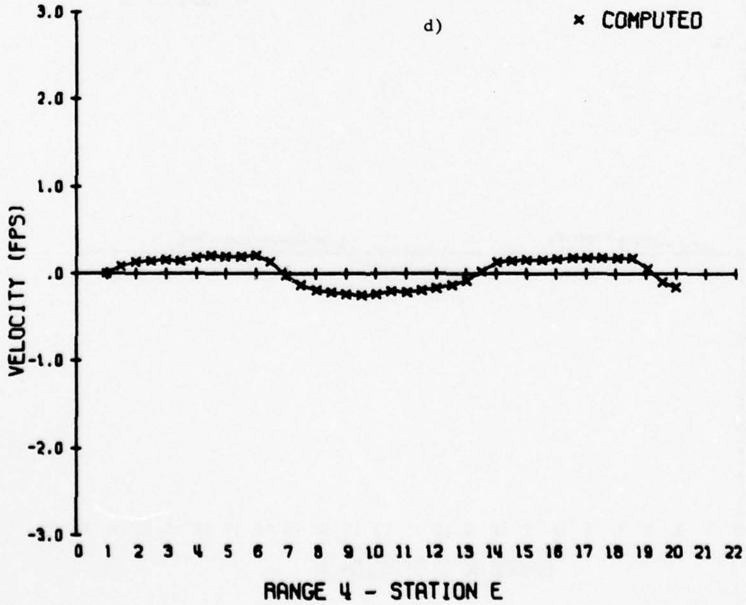
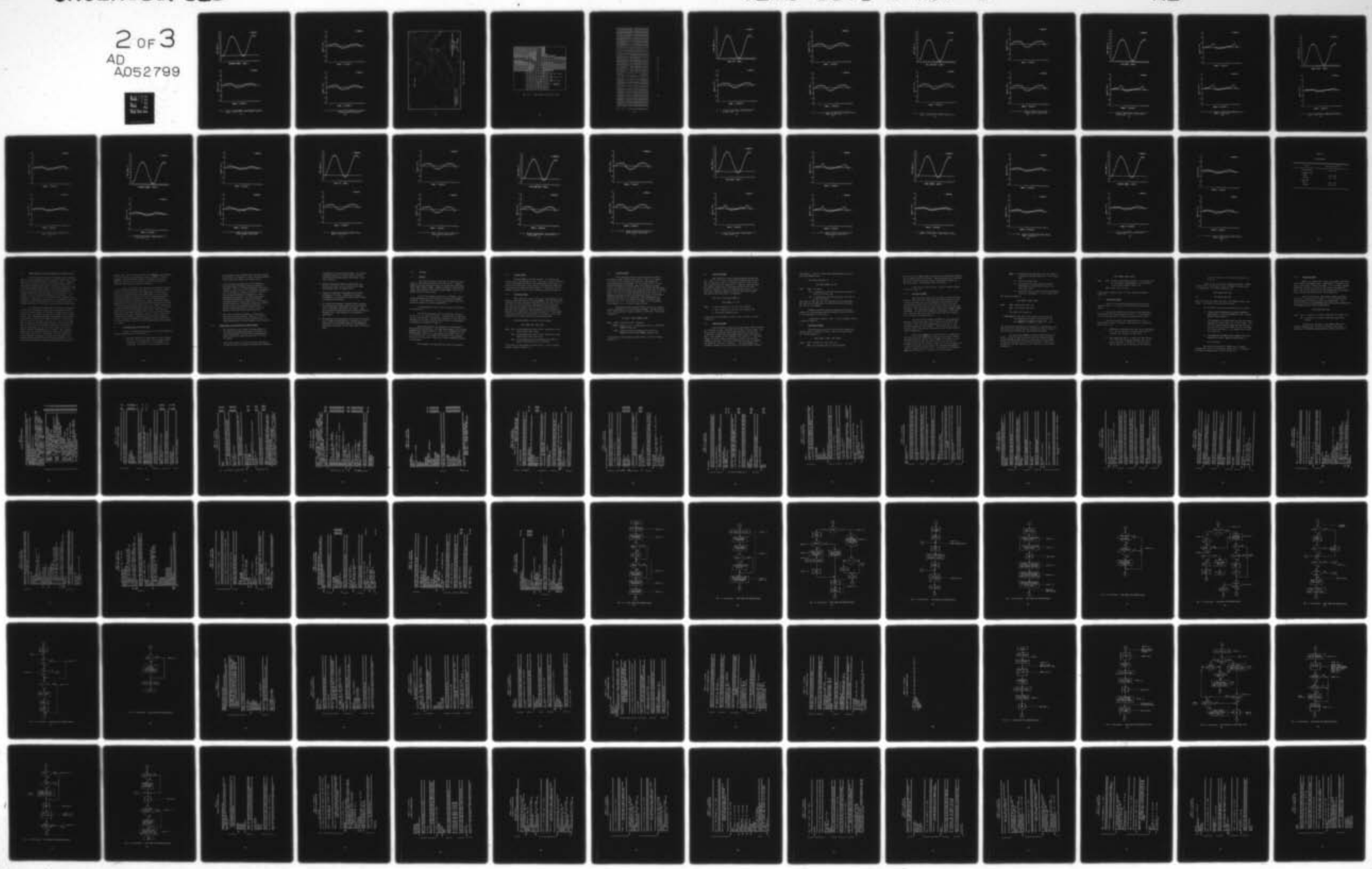
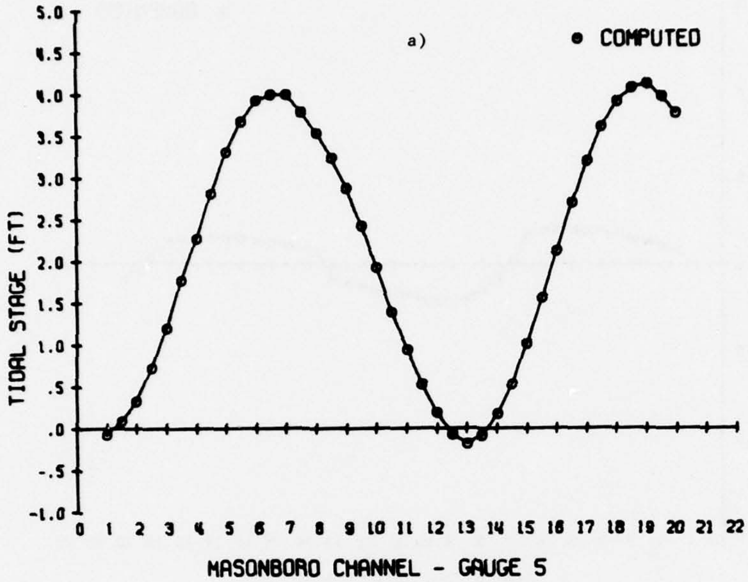


FIG. 4.37 (Continued) RESULTS OF RUN FOR NOVEMBER, 1964, SPRING TIDE FOR STATION 4, a) TIDE, b) VELOCITY WEST, c) VELOCITY CENTER, d) VELOCITY EAST

AD-A052 799 TRACOR INC AUSTIN TEX F/G 8/8
COMPARISON OF NUMERICAL AND PHYSICAL HYDRAULIC MODELS, MASONBOR--ETC(U)
JUN 77 R J CHEN, L A HEMBREE DACW72-72-C-0029
UNCLASSIFIED CERC-GITI-6-APP-3 NL

2 OF 3
AD
A052799





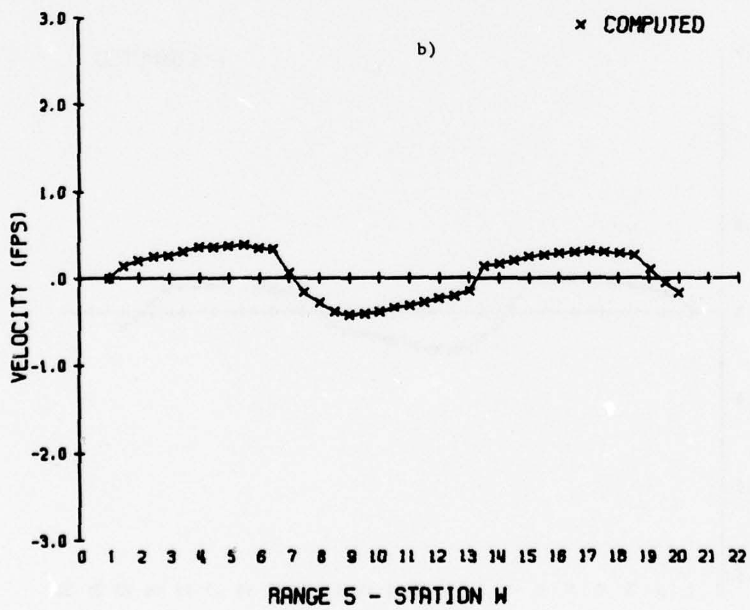
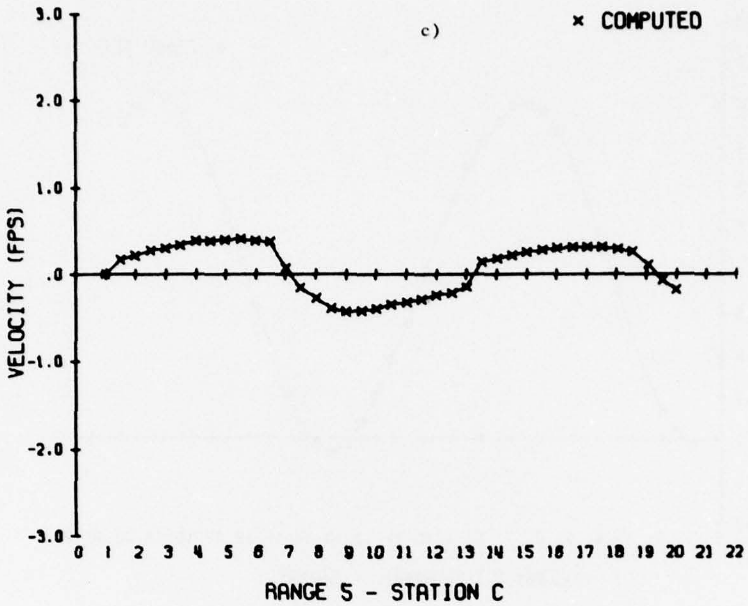


FIG. 4.38 RESULTS OF RUN FOR NOVEMBER, 1964, SPRING TIDE FOR STATION 5, a) TIDE, b) VELOCITY WEST, c) VELOCITY CENTER, d) VELOCITY EAST



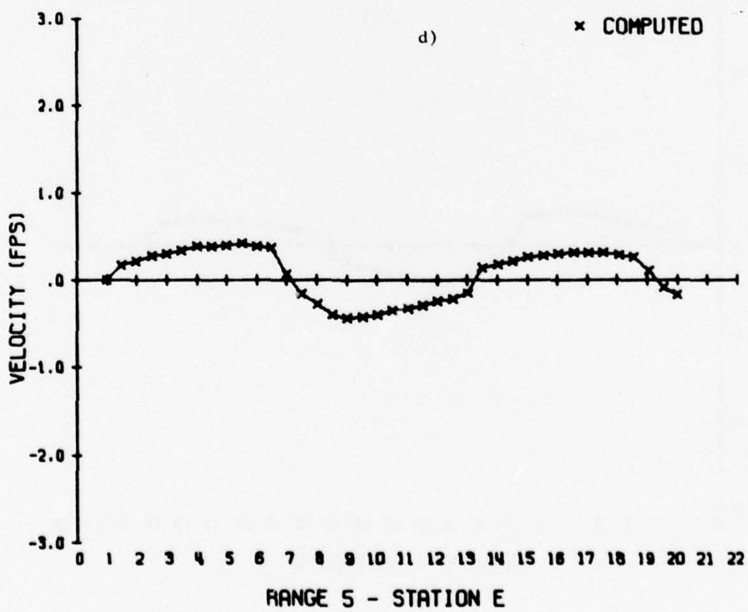


FIG. 4.38 (Continued) RESULTS OF RUN FOR NOVEMBER, 1964, SPRING TIDE FOR STATION 5, a) TIDE, b) VELOCITY WEST, c) VELOCITY CENTER, d) VELOCITY EAST

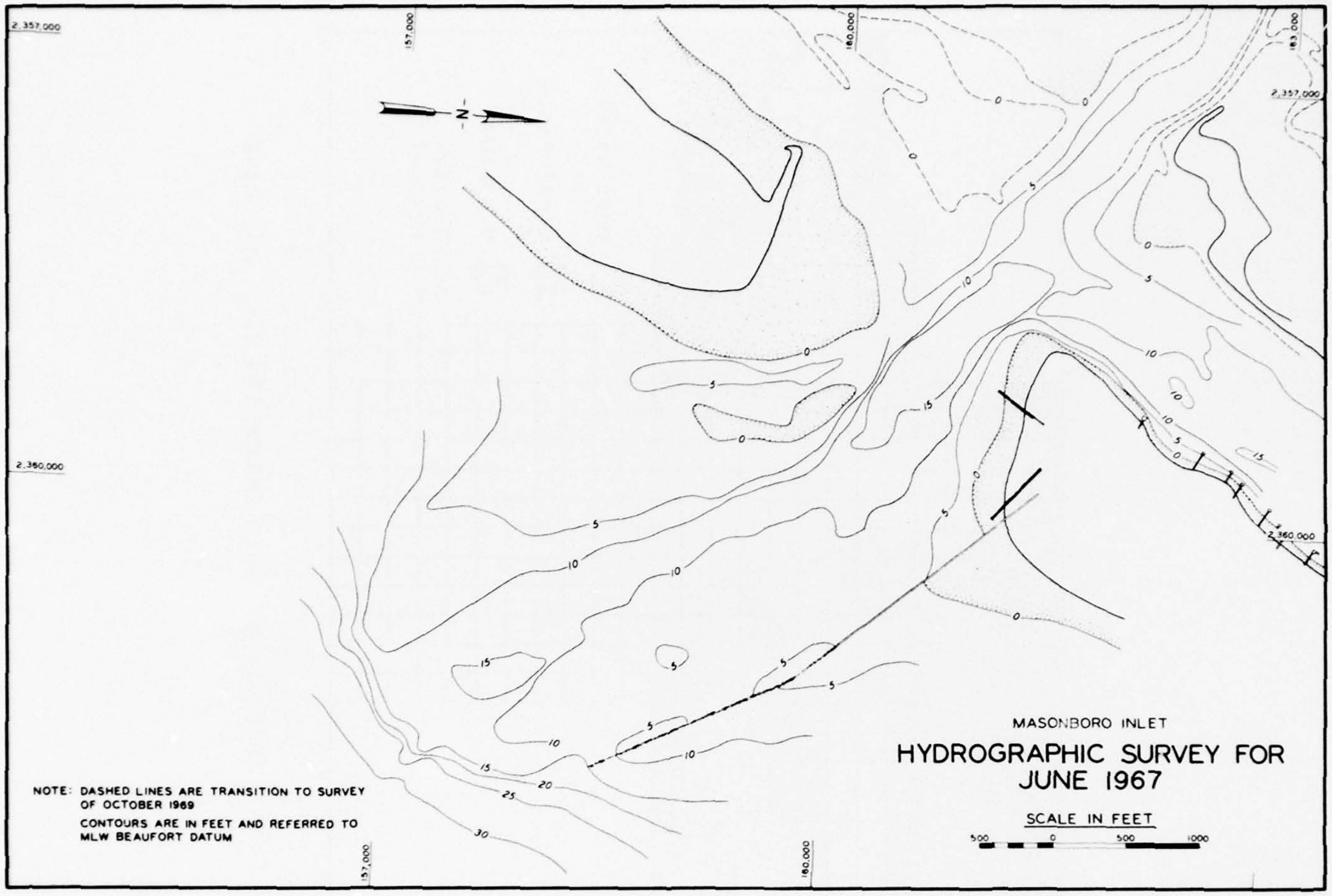


FIG. 4.39 JUNE 1967 SURVEY

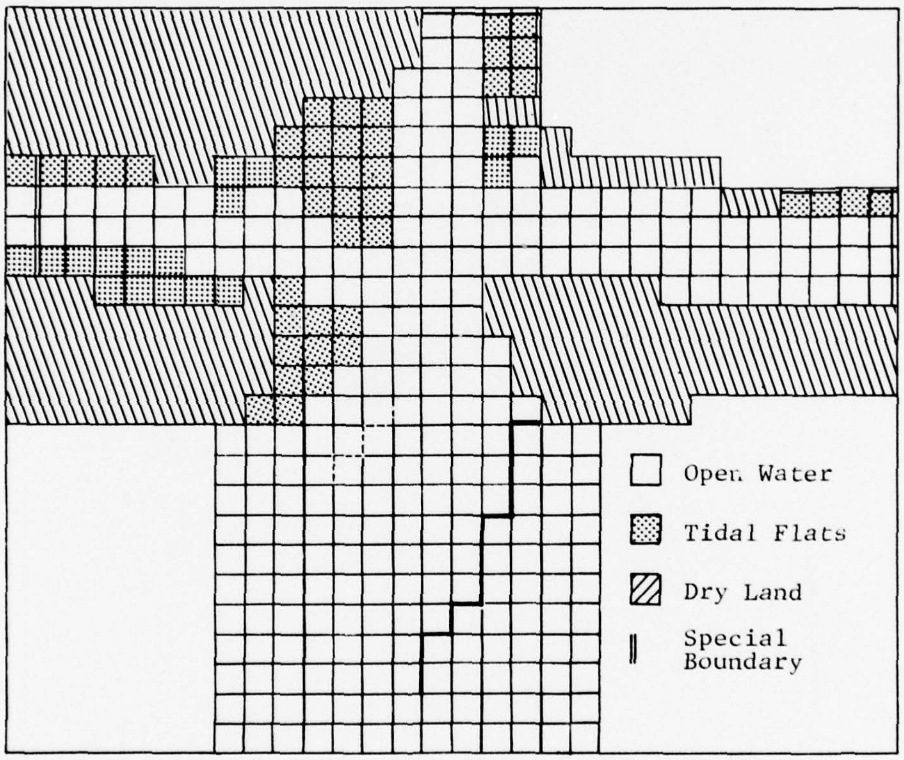
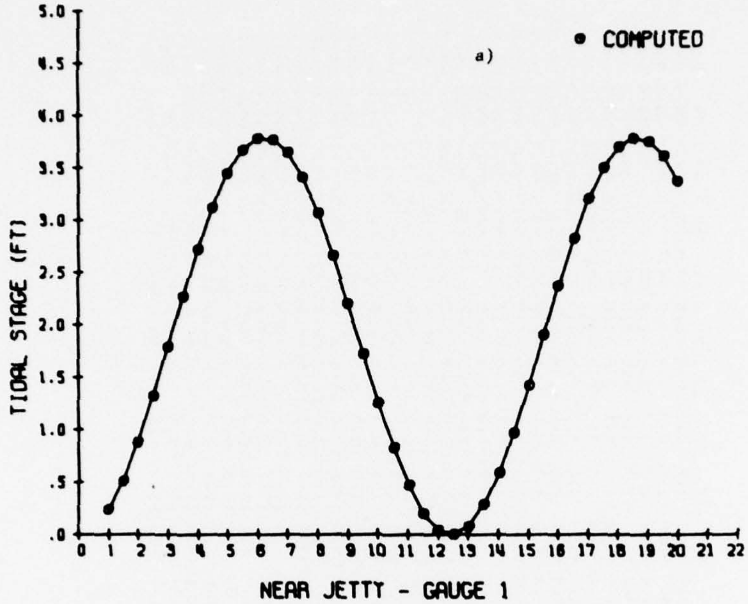


FIG. 4.40 GRID SYSTEM FOR JUNE 1967 CASE

I/J	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-1.8	3.0	4.0	-3.0	99.9	99.9	99.9	99.9	99.9
2	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-1.8	3.3	4.0	-3.0	99.9	99.9	99.9	99.9	99.9
3	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-1.7	3.3	4.6	-2.9	99.9	99.9	99.9	99.9	99.9
4	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-2.3	-1.3	4.6	4.1	-2.8	99.9	99.9	99.9	99.9	99.9
5	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-2.3	-1.0	5.1	3.8	-2.8	99.9	99.9	99.9	99.9	99.9
6	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	-2.0	-1.1	6.1	2.2	99.9	99.9	99.9	99.9	99.9	99.9
7	99.9	10.0	9.0	7.0	5.0	5.0	5.0	5.0	4.0	3.0	.7	99.9	99.9	99.9	99.9	-1.5	2.0	6.3	1.5	99.9	99.9	99.9	99.9	99.9	99.9
8	10.0	10.0	9.0	7.0	5.0	5.0	5.0	5.0	4.0	3.0	.7	99.9	99.9	99.9	99.9	-1.5	2.9	6.4	-1.5	-1.2	99.9	99.9	99.9	99.9	99.9
9	10.0	10.0	10.0	8.0	6.0	5.0	5.0	5.0	4.0	4.0	.7	-1.0	99.9	99.9	99.9	99.9	3.3	6.0	.5	-1.0	99.9	99.9	99.9	99.9	99.9
10	13.0	13.0	12.0	11.0	9.0	6.0	5.0	5.0	4.0	4.0	3.0	-.4	-1.0	-2.0	-2.0	-.5	3.0	5.6	.5	-1.5	-1.0	99.9	99.9	99.9	99.9
11	15.0	15.0	15.0	13.0	11.0	10.0	7.0	5.0	4.0	1.0	4.0	.5	-.4	-1.0	-1.0	.5	4.3	2.1	-1.0	-1.0	-.7	-1.5	99.9	99.9	99.9
12	11.0	11.0	10.0	10.0	11.0	10.0	11.0	7.0	4.0	1.0	2.0	4.0	.5	-.4	-.4	2.0	5.0	-1.0	-1.5	-.8	-.6	-2.5	99.9	99.9	99.9
13	14.0	14.0	10.0	9.0	9.0	8.0	10.0	10.0	9.0	5.0	1.0	1.0	4.0	3.0	3.0	5.0	6.0	-1.0	-1.5	-.7	-.5	-3.5	99.9	99.9	99.9
14	15.0	15.0	7.0	7.0	6.0	8.0	9.0	9.0	10.0	9.0	8.0	13.0	13.0	13.0	13.0	11.0	8.0	9.0	8.0	7.0	7.0	6.0	4.0	99.9	99.9
15	17.0	17.0	10.0	7.0	6.0	7.0	7.0	9.0	9.0	10.0	9.0	14.0	15.0	15.0	14.0	15.0	15.0	13.0	11.0	11.0	11.0	11.0	10.0	6.0	5.0
16	19.0	19.0	13.0	10.0	9.0	5.0	6.0	6.0	9.0	8.0	9.0	9.0	8.0	8.0	5.0	8.0	12.0	11.0	6.0	5.0	6.0	6.0	7.0	8.0	8.0
17	20.0	20.0	18.0	13.0	10.0	7.0	5.0	5.0	5.0	5.0	5.0	4.0	1.0	1.9	99.9	99.9	11.0	10.0	7.0	-1.0	-1.0	99.9	-1.4	-1.5	-1.6
18	20.0	20.0	20.0	15.0	13.0	10.0	8.0	5.0	4.0	3.0	.5	1.0	99.9	99.9	99.9	99.9	11.0	9.0	6.0	1.3	-2.0	99.9	-1.9	-1.9	-2.0
19	20.0	20.0	20.0	15.0	13.0	11.0	10.0	8.0	5.0	3.0	.5	99.9	99.9	99.9	99.9	99.9	11.0	10.0	8.0	99.9	99.9	99.9	99.9	99.9	99.9
20	20.0	20.0	20.0	15.0	13.0	11.0	10.0	8.0	5.0	3.0	.5	99.9	99.9	99.9	99.9	99.9	10.0	12.0	11.0	99.9	99.9	99.9	99.9	99.9	99.9
21	20.0	20.0	20.0	15.0	13.0	11.0	10.0	8.0	5.0	3.0	.5	99.9	99.9	99.9	99.9	99.9	11.6	3.8	11.0	99.9	99.9	99.9	99.9	99.9	99.9
22	99.9	20.0	20.0	15.0	13.0	11.0	10.0	8.0	5.0	3.0	.5	99.9	99.9	99.9	99.9	99.9	11.8	8.7	10.0	99.9	99.9	99.9	99.9	99.9	99.9
23	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	3.5	12.3	12.4	9.0	99.9	99.9	99.9	99.9	99.9	99.9
24	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	5.6	13.2	11.9	8.0	99.9	99.9	99.9	99.9	99.9	99.9
25	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	6.4	15.8	10.8	99.9	99.9	99.9	99.9	99.9	99.9	99.9
26	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	5.6	17.3	6.0	99.9	99.9	99.9	99.9	99.9	99.9	99.9
27	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	6.3	17.6	6.0	-.5	99.9	99.9	99.9	99.9	99.9	99.9
28	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	8.0	17.0	8.0	-.5	99.9	99.9	99.9	99.9	99.9	99.9
29	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	8.0	13.5	1.5	-.5	99.9	99.9	99.9	99.9	99.9	99.9
30	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9	8.0	13.0	2.0	-.5	99.9	99.9	99.9	99.9	99.9	99.9

FIG. 4.41 DEPTHS FOR JUNE 1967 CASE



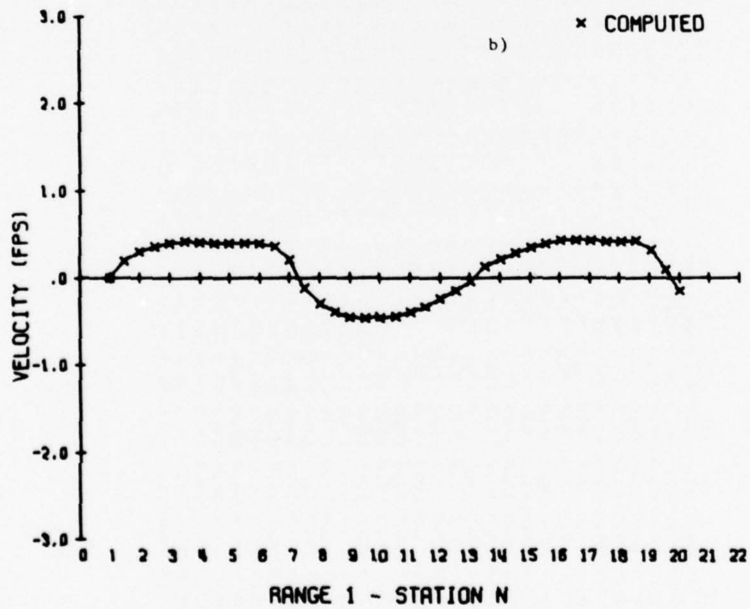
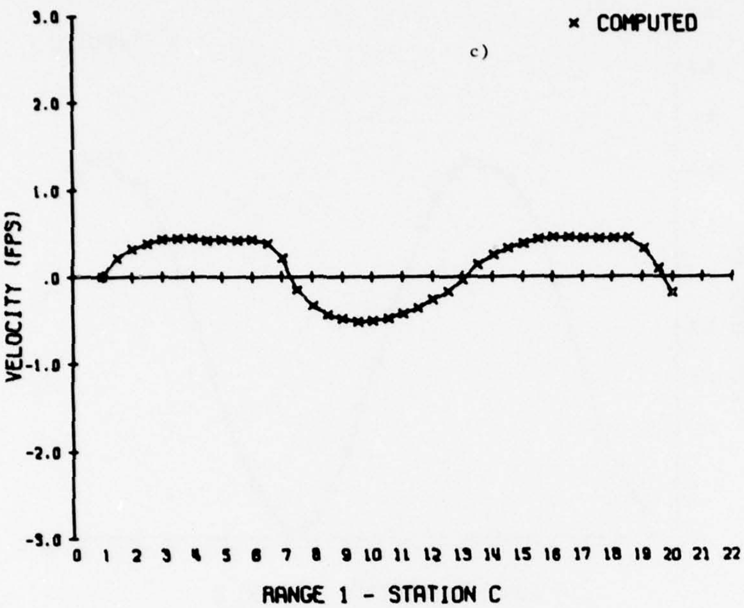


FIG. 4.42 RESULTS OF RUN FOR JUNE 1967, MEAN TIDE FOR STATION 1, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



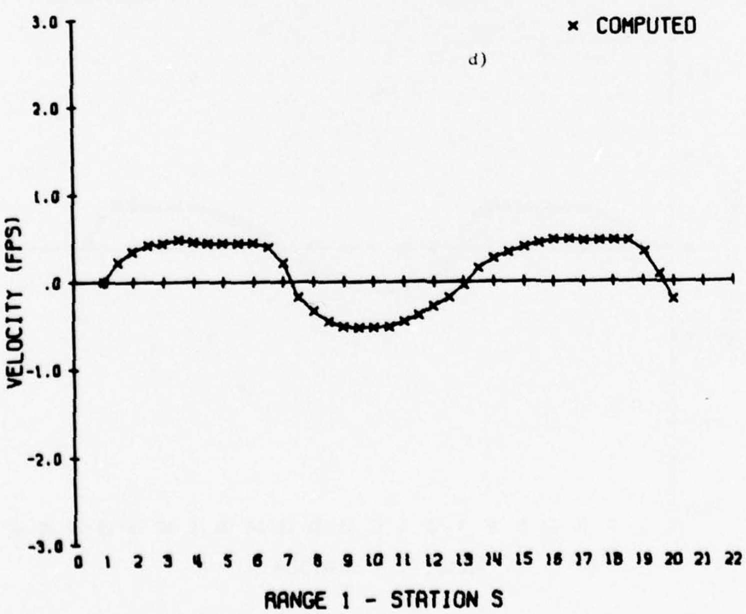
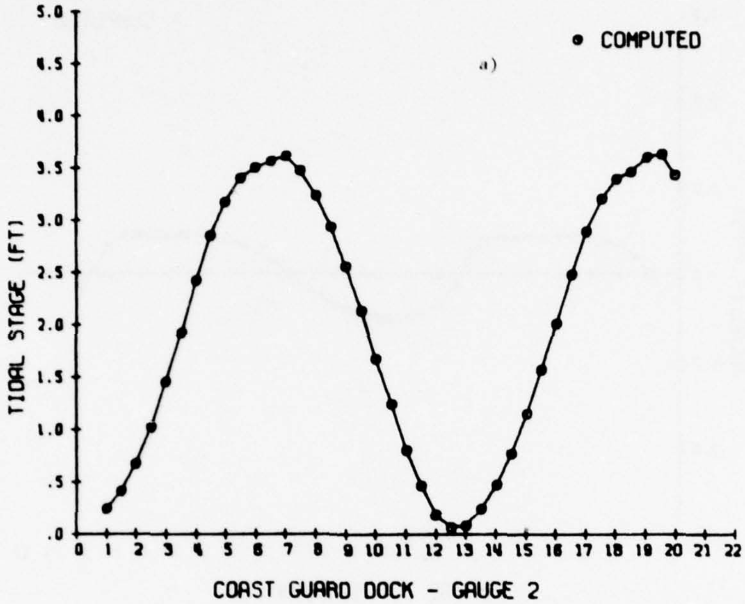


FIG. 4.42 (Continued) RESULTS OF RUN FOR JUNE 1967, MEAN TIDE FOR STATION 1, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



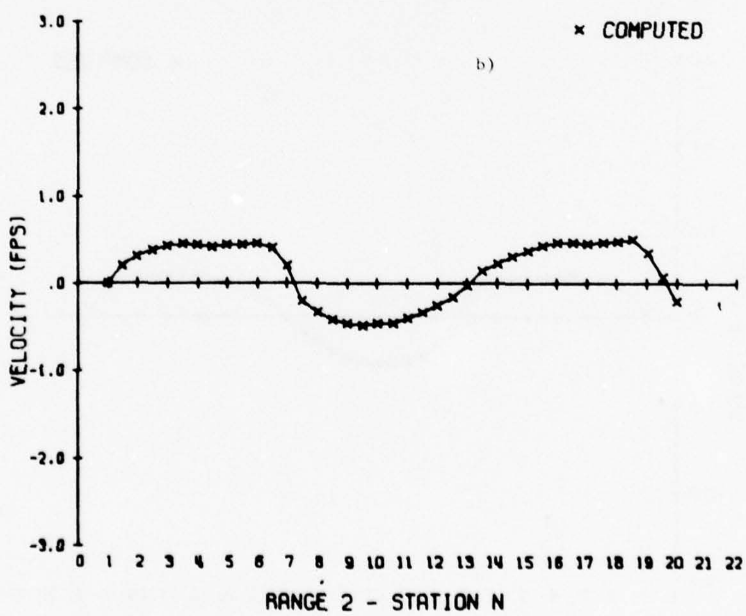
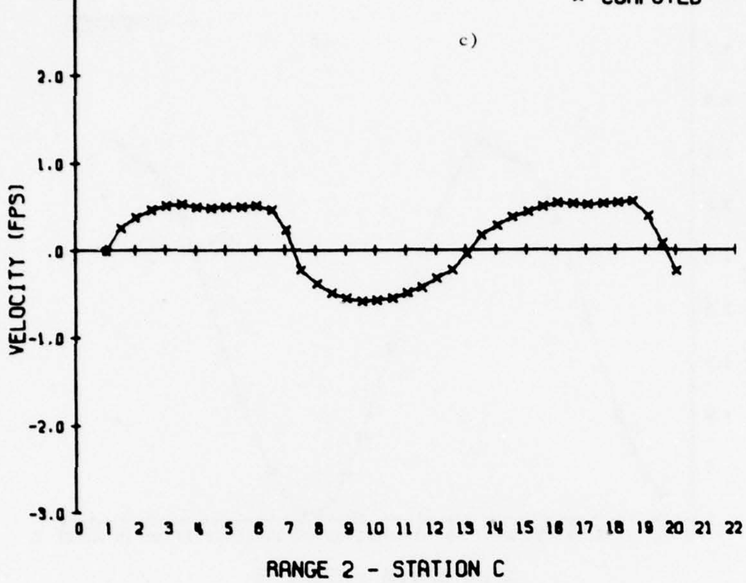


FIG. 4.43 RESULTS OF RUN FOR JUNE 1967, MEAN TIDE FOR STATION 2, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



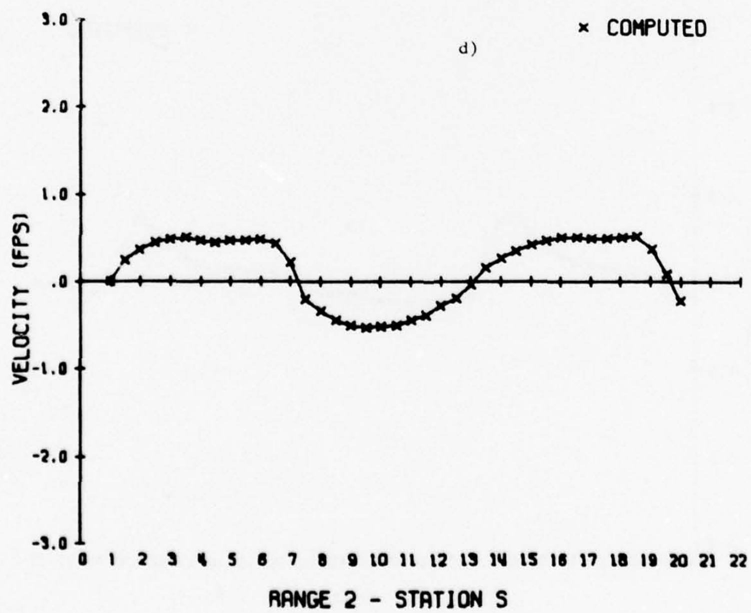


FIG. 4.43 (Continued) RESULTS OF RUN FOR JUNE 1967, MEAN TIDE FOR STATION 2, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH

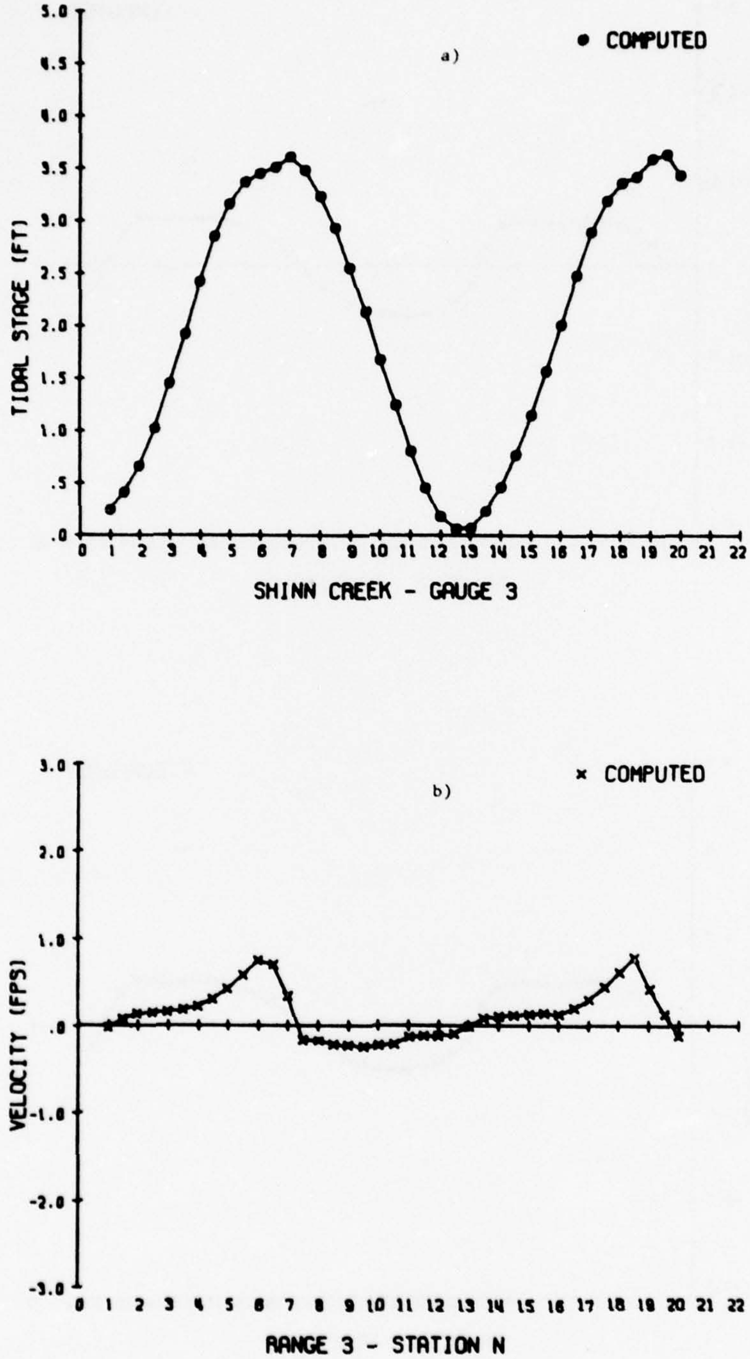
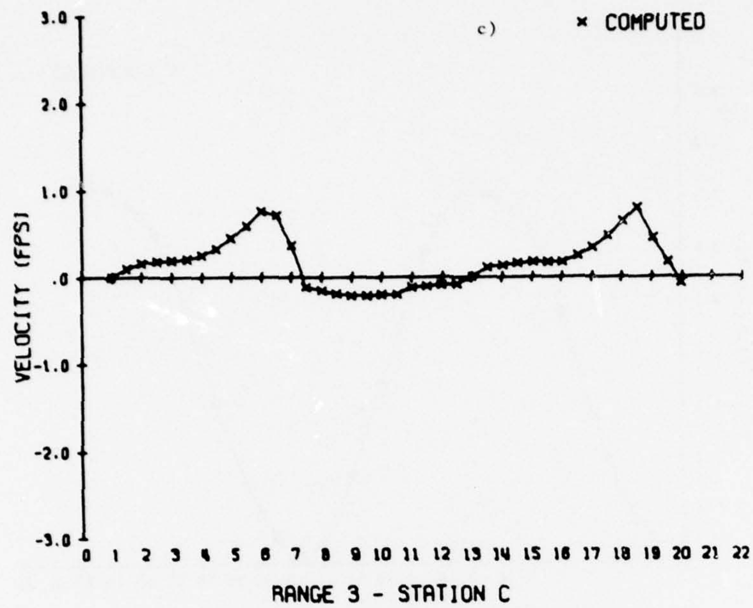


FIG. 4.44 RESULTS OF RUN FOR JUNE 1967, MEAN TIDE FOR STATION 3, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



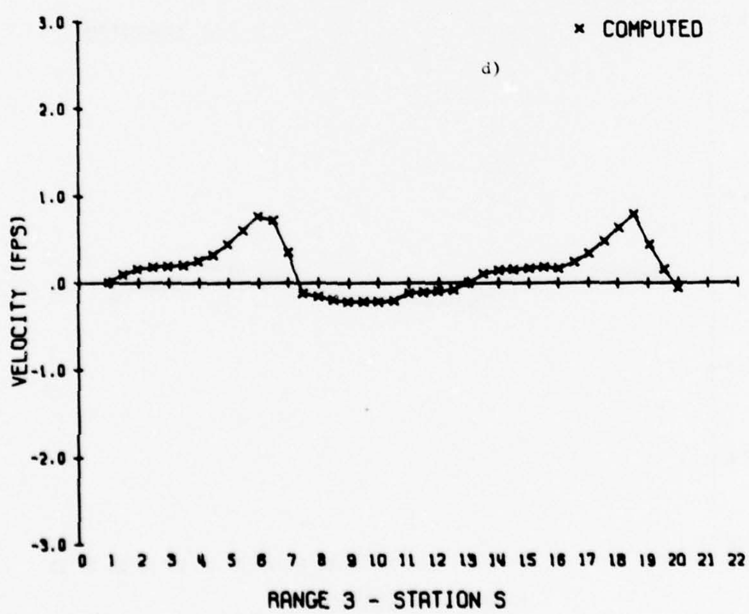
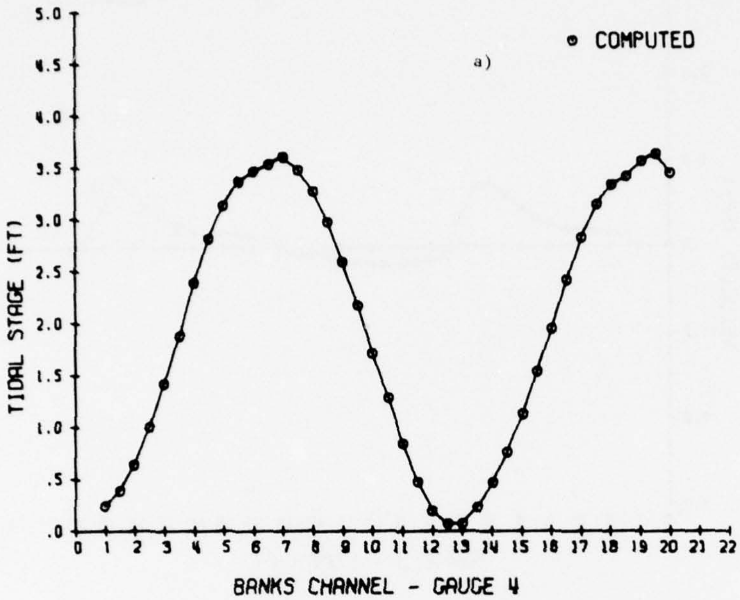


FIG. 4.44 (Continued) RESULTS OF RUN FOR JUNE 1967, MEAN TIDE FOR STATION 3, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



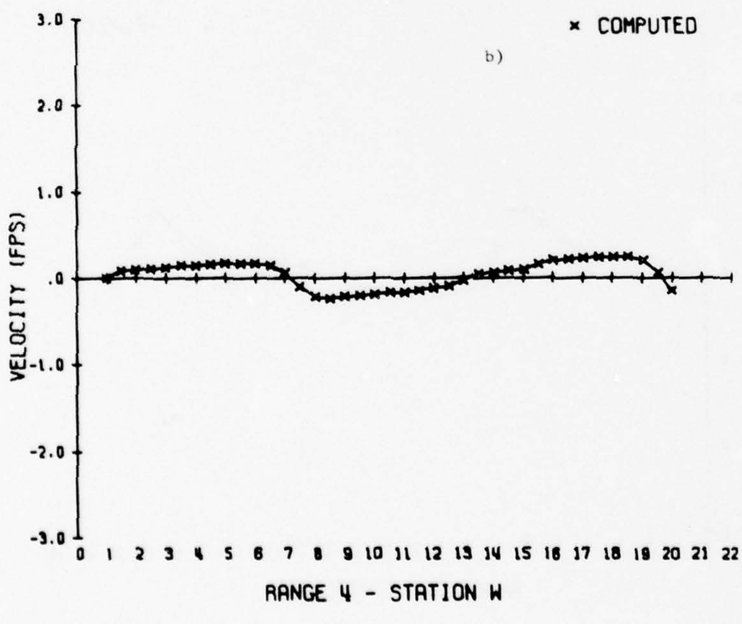
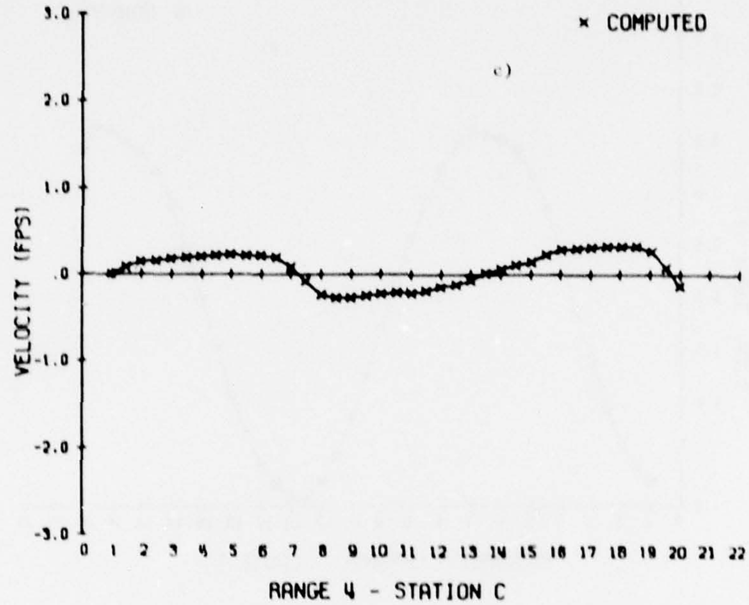


FIG. 4.45 RESULTS OF RUN FOR JUNE 1967, MEAN TIDE FOR STATION 4, a) TIDE, b) VELOCITY WEST, c) VELOCITY CENTER, d) VELOCITY EAST



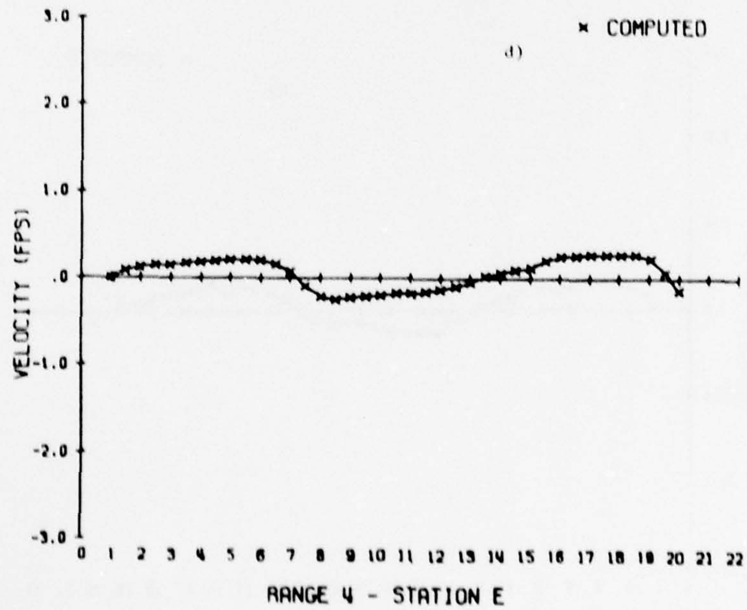
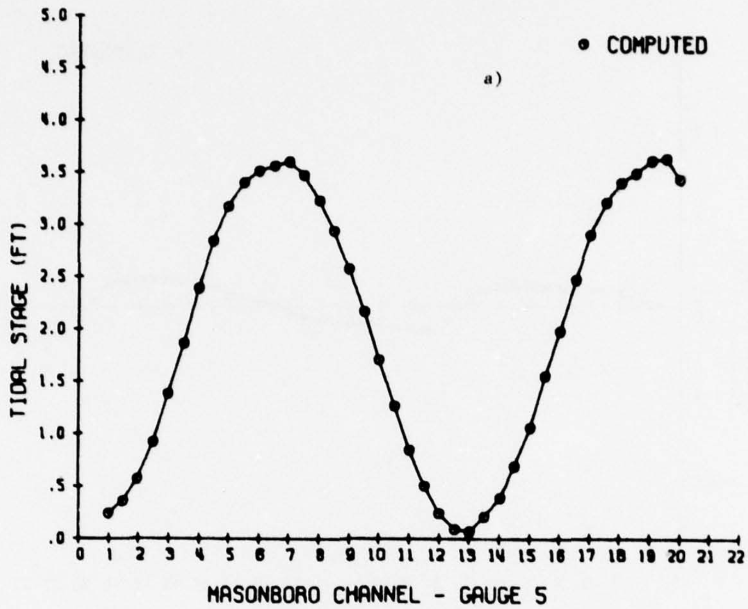


FIG. 4.45 (Continued) RESULTS OF RUN FOR JUNE 1967, MEAN TIDE FOR STATION 4, a) TIDE, b) VELOCITY WEST, c) VELOCITY CENTER, d) VELOCITY EAST



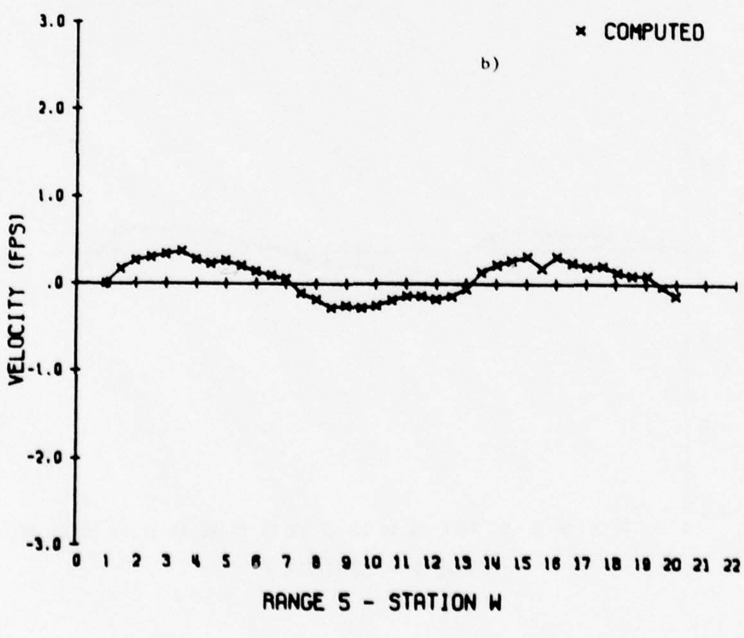
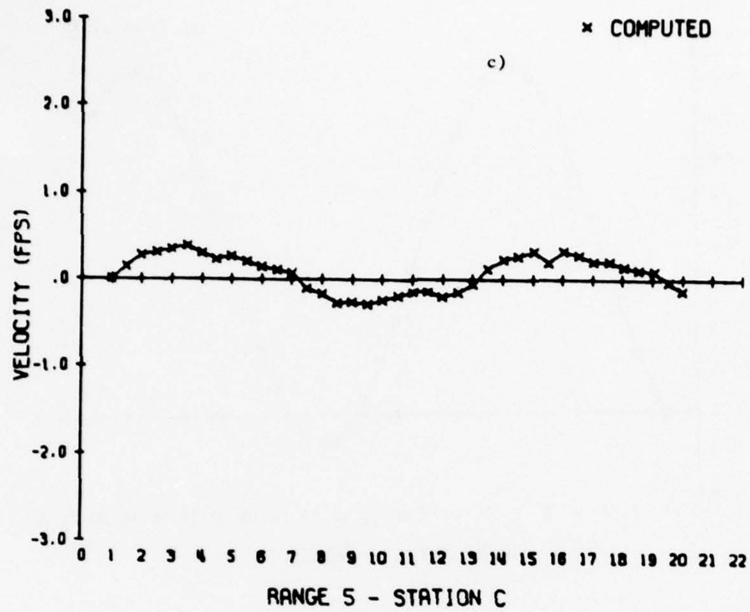


FIG. 4.46 RESULTS OF RUN FOR JUNE 1967, MEAN TIDE FOR STATION 5, a) TIDE, b) VELOCITY WEST, c) VELOCITY CENTER, d) VELOCITY EAST



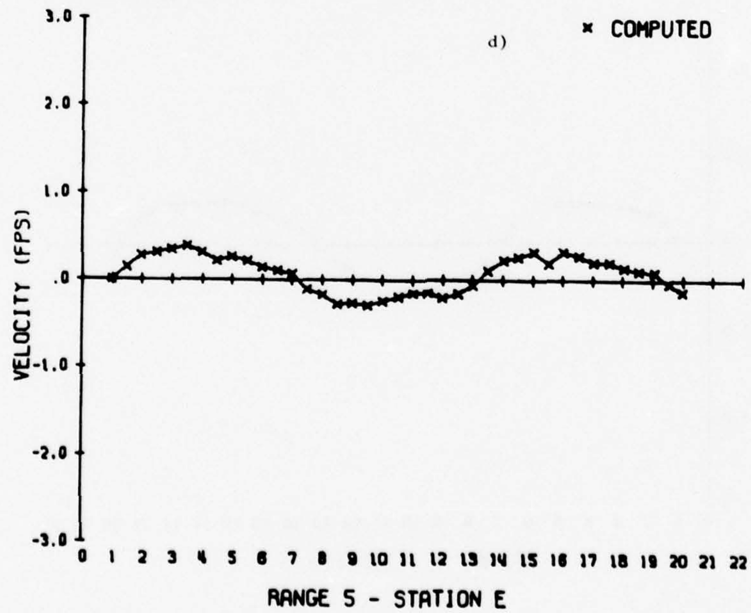
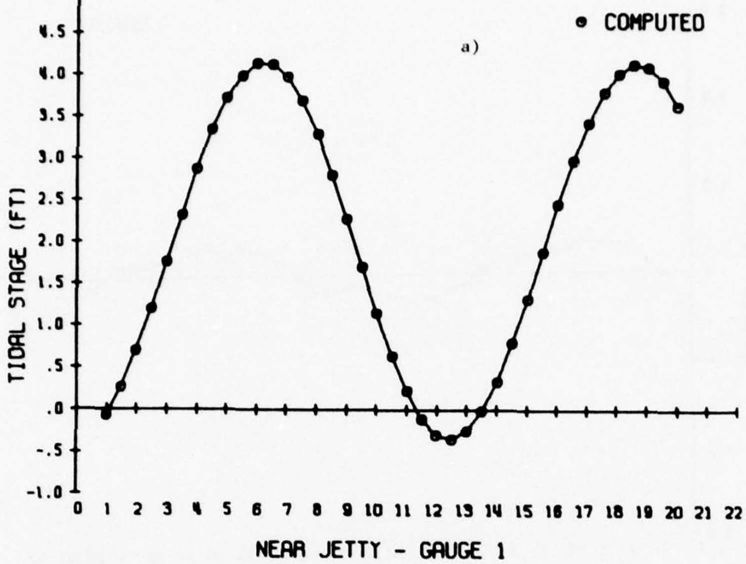


FIG. 4.46 (Continued) RESULTS OF RUN FOR JUNE 1967, MEAN TIDE FOR STATION 5, a) TIDE, b) VELOCITY WEST, c) VELOCITY CENTER, d) VELOCITY EAST



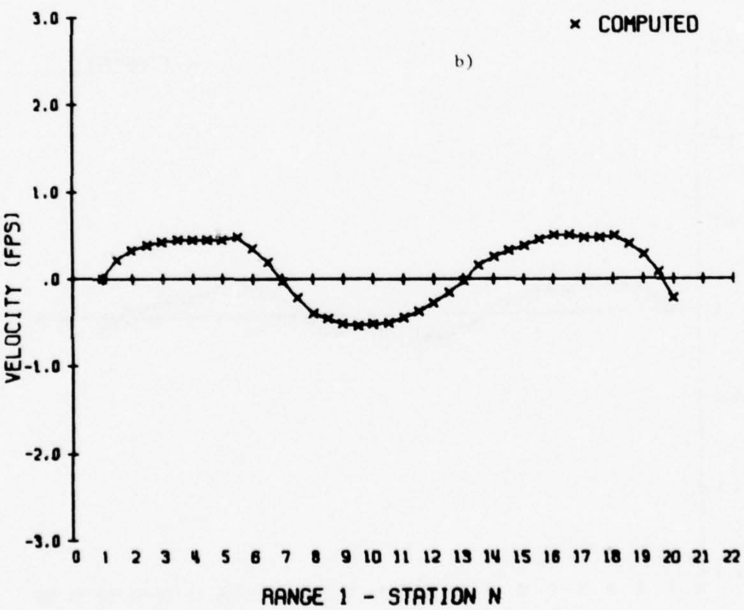
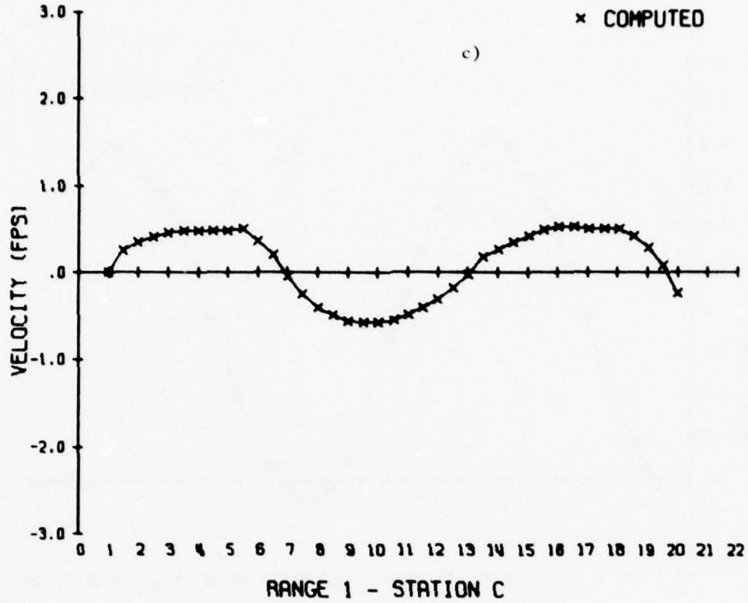


FIG. 4.47 RESULTS OF RUN FOR JUNE 1967, SPRING TIDE FOR STATION 1, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



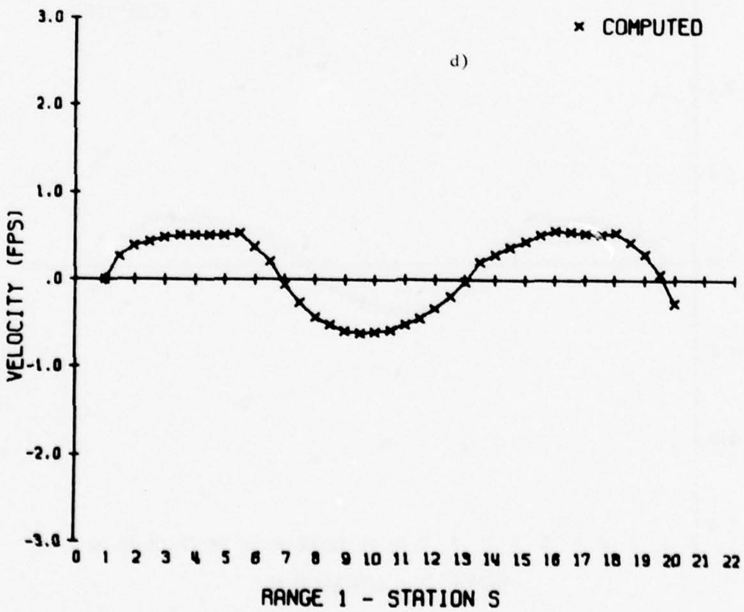
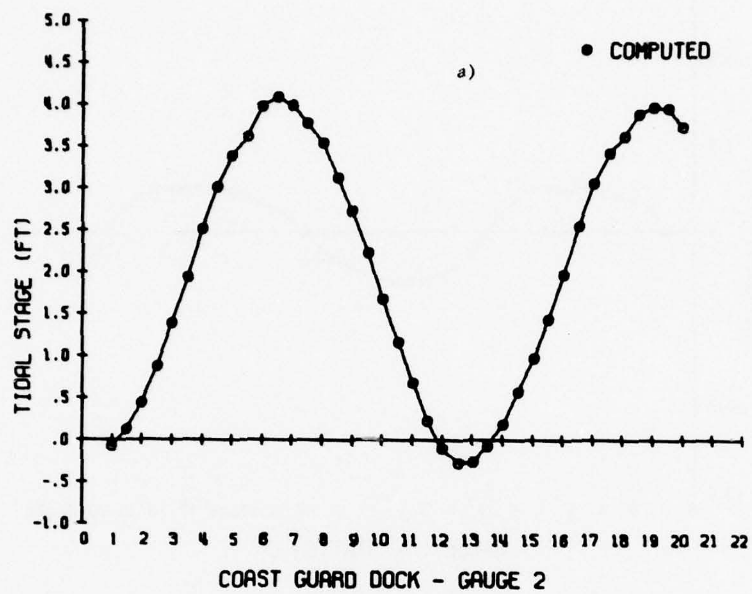


FIG. 4.47 (Continued) RESULTS OF RUN FOR JUNE 1967, SPRING TIDE FOR STATION 1, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



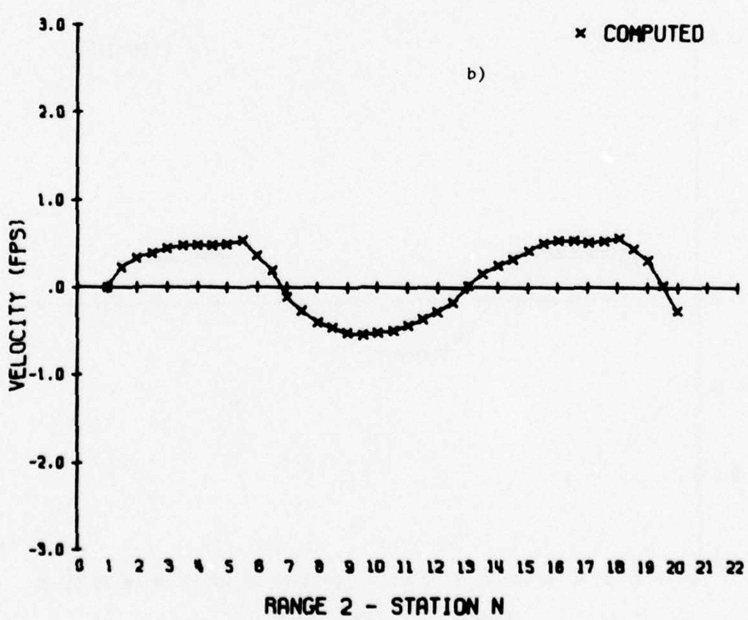
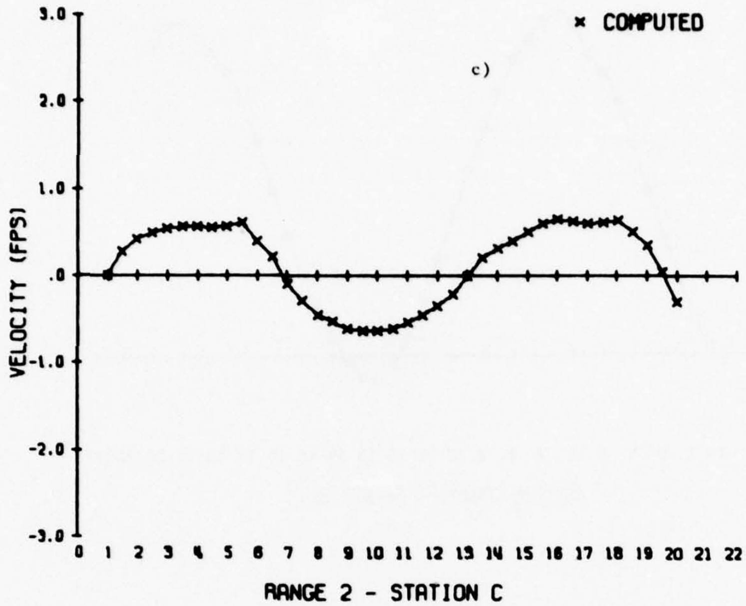


FIG. 4.48 RESULTS OF RUN FOR JUNE 1967, SPRING TIDE FOR STATION 2, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



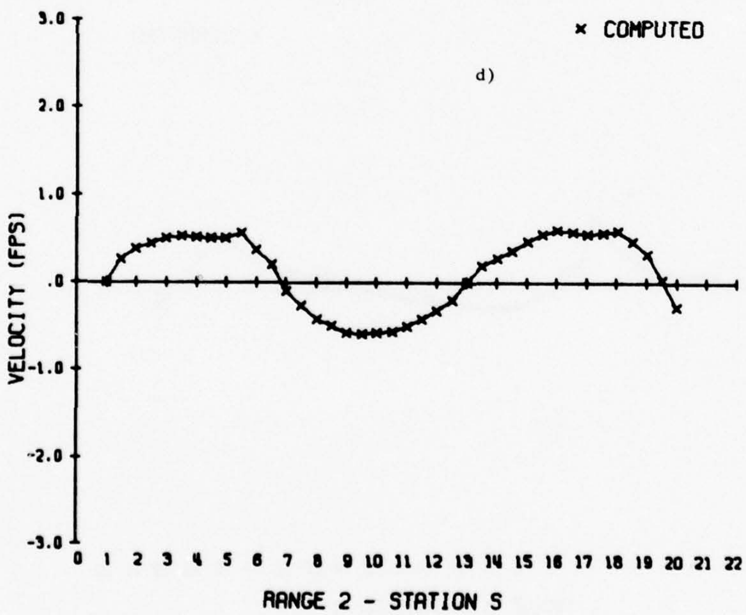
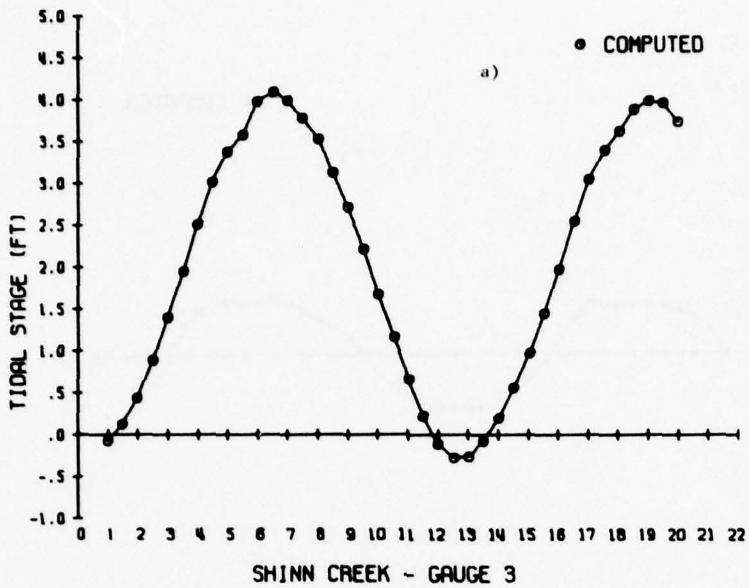


FIG. 4.48 (Continued) RESULTS OF RUN FOR JUNE 1967, SPRING TIDE FOR STATION 2, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



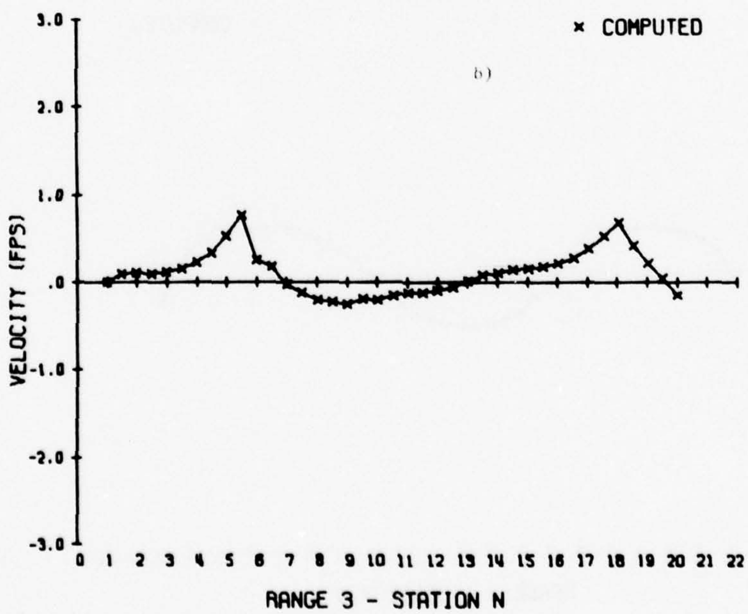
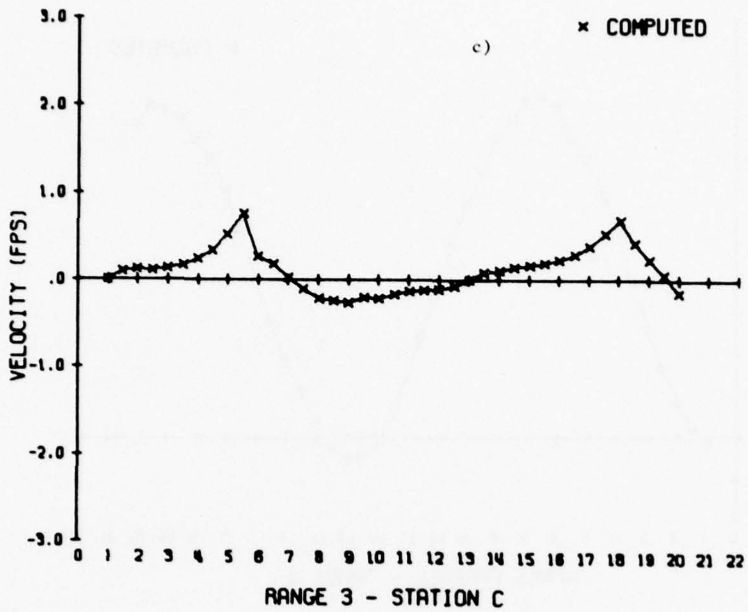


FIG. 4.49 RESULTS OF RUN FOR JUNE 1967, SPRING TIDE FOR STATION 3, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



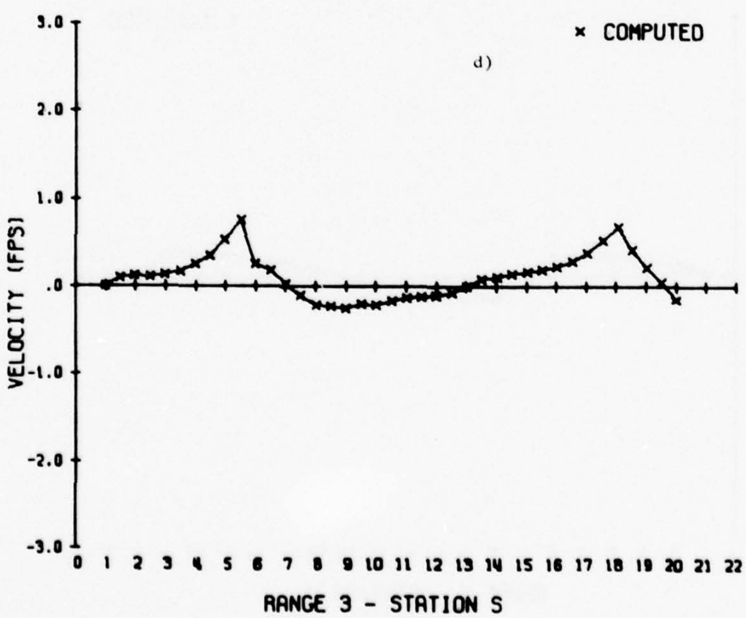
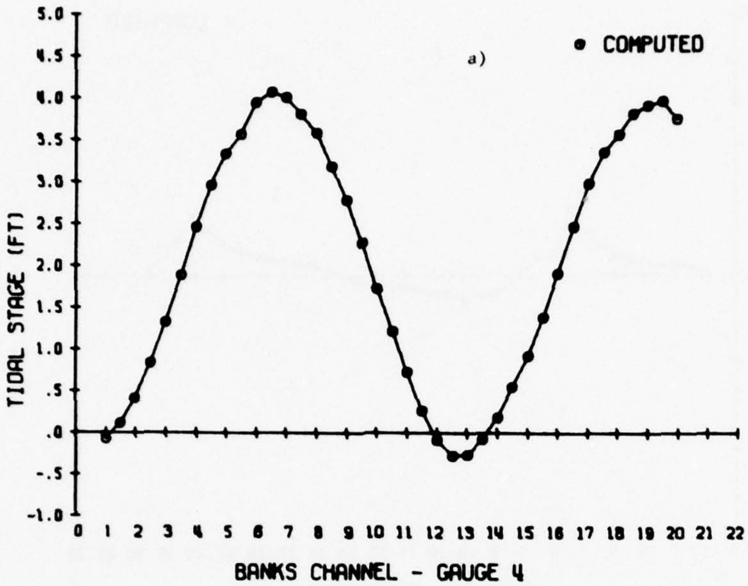


FIG. 4.49 (Continued) RESULTS OF RUN FOR JUNE 1967, SPRING TIDE FOR STATION 3, a) TIDE, b) VELOCITY NORTH, c) VELOCITY CENTER, d) VELOCITY SOUTH



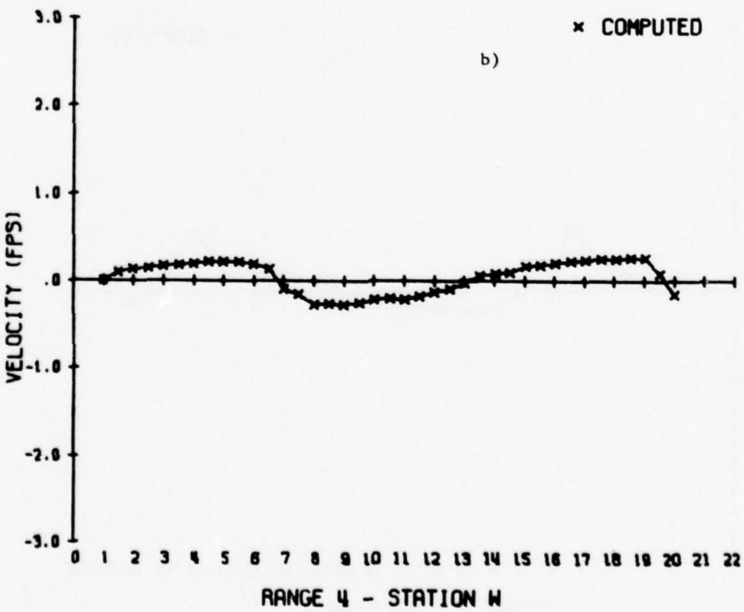
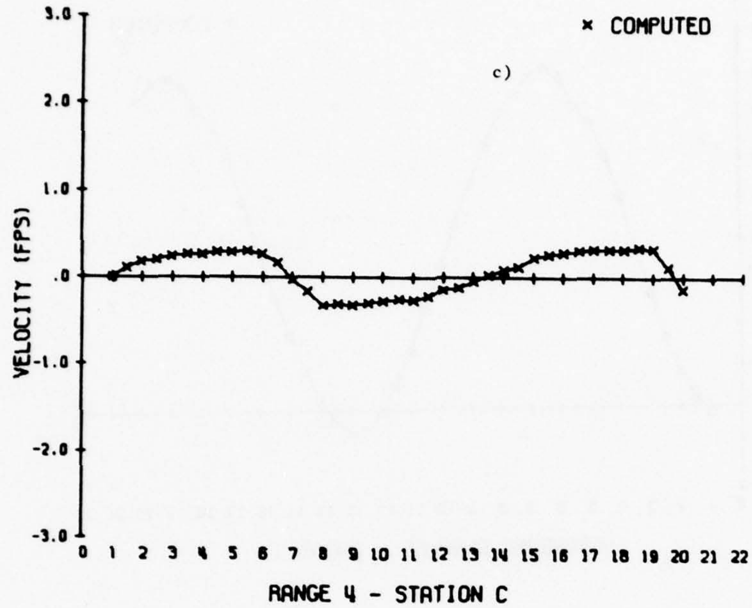


FIG. 4.50 RESULTS OF RUN FOR JUNE 1967, SPRING TIDE FOR STATION 4, a) TIDE, b) VELOCITY WEST, c) VELOCITY CENTER, d) VELOCITY EAST



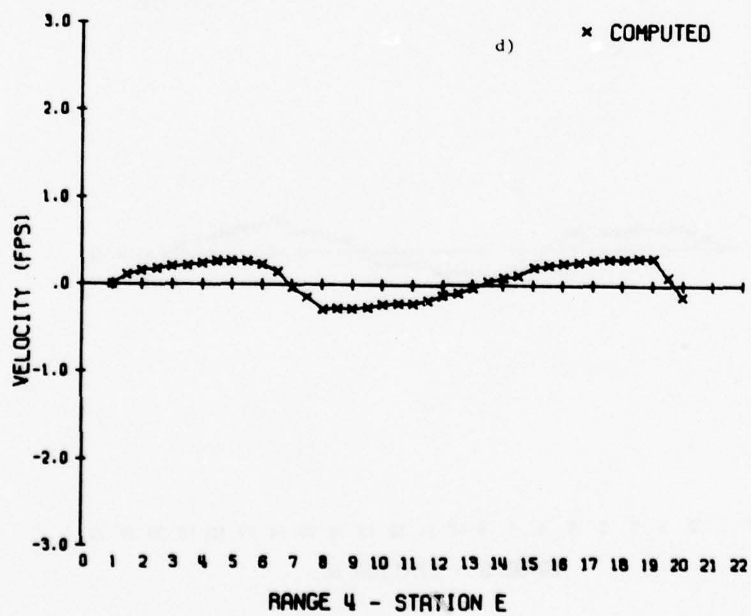
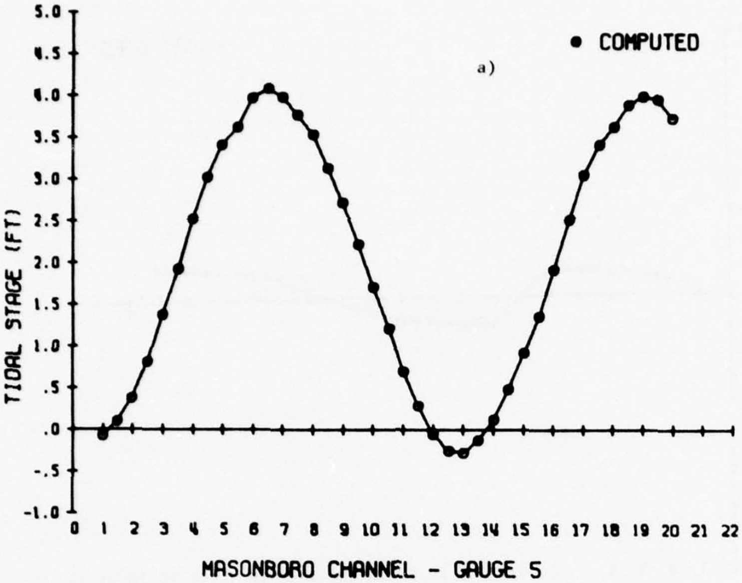


FIG. 4.50(Continued) RESULTS OF RUN FOR JUNE 1967, SPRING TIDE FOR STATION 4, a) TIDE, b) VELOCITY WEST, c)VELOCITY CENTER, d) VELOCITY EAST



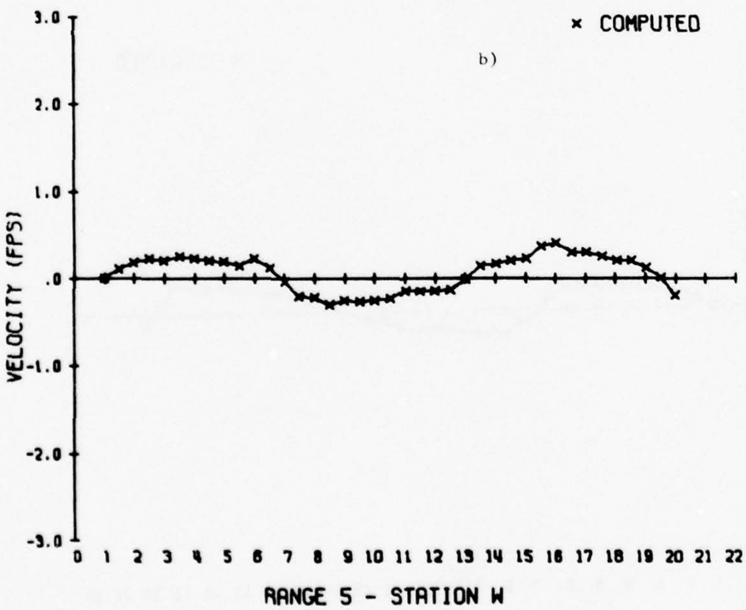
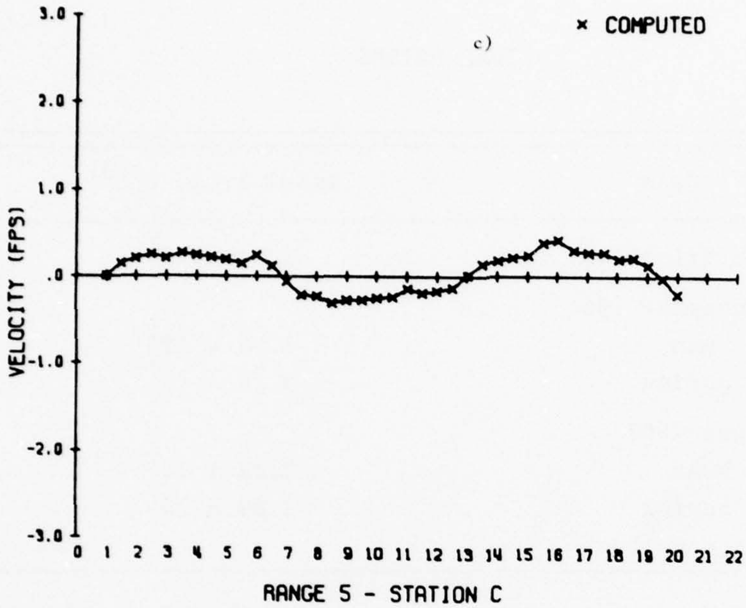


FIG. 4.51 RESULTS OF RUN FOR JUNE 1967, SPRING TIDE FOR STATION 5, a) TIDE, b) VELOCITY WEST, c) VELOCITY CENTER, d) VELOCITY EAST



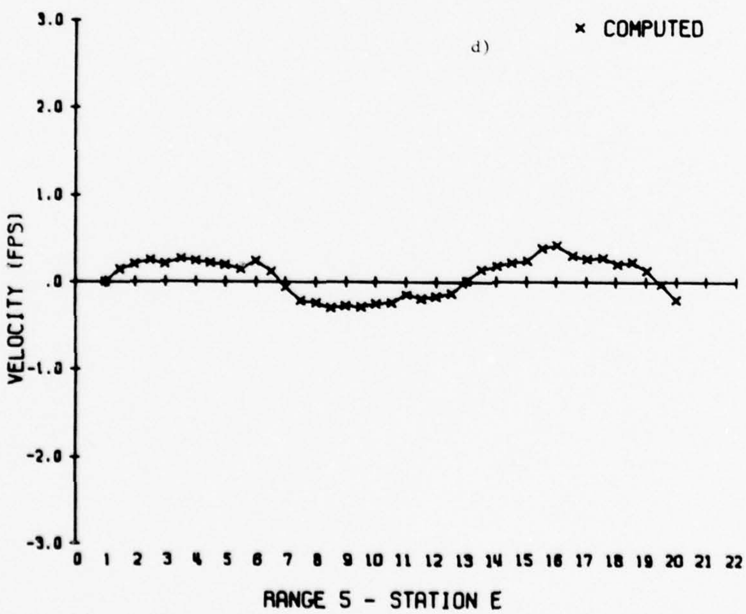


FIG. 4.51(Continued) RESULTS OF RUN FOR JUNE 1967, SPRING TIDE FOR STATION 5, a) TIDE, b) VELOCITY WEST, c) VELOCITY CENTER, d) VELOCITY EAST

TABLE 4.2

TIDAL PRISMS

Case	Tidal Prism (ft^3)
Verification	2.81×10^7
November 1964	
Mean	1.94×10^7
Spring	3.24×10^7
June 1967	
Mean	2.82×10^7
Spring	3.89×10^7

4.3 Applicability of Present Approach to Masonboro Inlet

The verification runs for 12 September 1969 establish that the present approach yields satisfactory predictions of most aspects of the tidal hydraulics for Masonboro Inlet. Computed tidal stage variation and intratidal current phasing agreed reasonably with the measured data. The most serious discrepancy was between computed and measured tidal current magnitudes. The source for this difficulty appears to be an underspecified boundary condition, associated with the open-end nature of the modeled system, as discussed elsewhere in this report. In addition, a close analysis of the empirical data indicates the influence of dramatic accelerations (e.g., transients, wind stress, longshore currents) which could not be included in the model simulations because of the lack of necessary observational data.

The most serious limitation to applying a computation of this type to Masonboro Inlet is not the theoretical basis, nor is it the performance of the model as gauged against observations, but it is rather the practical economics of the application. The confined physiography of Masonboro Inlet required a relatively fine spatial step (300 feet) merely to resolve the gross features of the inlet (see Fig. 4.2). It is our opinion that an even more refined network would be required to resolve lateral shears and cross-channel circulations. Even with the 300-foot representation, the time step necessary to satisfy linear stability requirements was 5 seconds. This entailed a running time on the UNIVAC 1108 of 32 minutes per simulated (12.4 hour) tidal cycle. The expense of such a computation clearly delimits the amount of testing and adjusting. It is doubtful that adopting an absolutely stable implicit or alternating implicit-explicit time step would materially improve the

running time. This is because, apart from stability considerations, the time step must be restrained in order to achieve a required accuracy. The detailed physiography of Masonboro Inlet would appear to demand as nearly as small a time step to maintain sufficient accuracy as to maintain stability.

If it is necessary that detailed cross-channel structure in the currents be resolved, then there is no alternative (numerically) to a fine-grid and costly computation. In many cases however it would appear that the channel-mean current and water level would suffice. In this case a more economical and equally satisfactory approach would appear to be the one-dimensional section-mean model as described, for example, in Dronkers (1964). Application of this type of computation by Harleman and Lee (1969) to American estuaries demonstrates its value. Simple modification to permit branching and multiple boundaries would allow its application directly to the Masonboro Inlet system. This approach would carry the bonus of obviating the need for a special boundary condition at the inlet boundaries, since it could be readily and economically extended to encompass the entire coastal network system. From this standpoint, development of such a model might be a useful adjunct even if the full two-dimensional representation is required.

4.4 Recommendations for Future Work

Specific recommendations pertaining to the two-dimensional model of this project are listed below.

1. The model should be verified against a more complete data set, or at least a data set for which external factors can be accounted for or established to be negligible. The effect of system "memory" cannot

be overlooked in this regard: events of the preceding several days can continue to cause transient responses, even if the causal agent is no longer active.

2. A more detailed investigation of the influence of the "unspecified" boundary conditions should be conducted. There is a strong possibility that this could be a source of error in the present computations, and this possibility should be explored. The conjecture that the boundary conditions are underdetermined when applied to an open system needs a thorough investigation. Alternative boundary conditions should be given some study. The present approach may prove to be entirely adequate if applied in the vicinity of a kinematical nodal point. Physical model studies may be useful in investigations of this and other related questions.

3. More study of numerical behavior of tidal flats hydraulics would be welcome. There are evidently some serious problems in large inundated areas. A better formulation of bottom friction may be required.

4.5 Suggestions for Application to Other Systems

1. Best application can be made to systems which can be completely encompassed by model calculation and whose physiography is sufficiently uncomplicated to permit large space steps (on the order of 10^3 feet or larger).

2. Tidal flats appear to be very sensitive numerically, in that waves can be excited by draining and inundating

cells which can corrupt the solution. If in doubt, a smaller time step or perhaps larger friction coefficients should be used in such areas. (See also 5, below.)

3. Initial calculations should be made using short duration runs only so as to catch bugs in input data without waste of computer time.

4. Validity of approach is dependent upon relative shallowness of system. Application to deeper waterbodies or to systems with deep reaches should be approached cautiously.

5. If a tidal-flat includes a special (open) boundary, the adjacent cells should flood before the boundary cell, or numerical instability can result. This can be handled by making the water depths successively shallower (by a small amount) as the boundary is approached.

6. The magnitude and distribution of friction coefficients can be roughly established very economically by operating first a "coarse-grid" model of the embayment, with space steps three to five times that of the final model.

5.0 PROGRAMS

5.1 General

The various programs comprising the Masonboro Inlet model are presented and discussed in this section. Program MASON is the main program and uses the subroutines FLAGS, MOD, TRACE, WIND, TIDAL2, PRINT1, SEARCH, and WFILE. The programs require approximately 2500 decimal words (60000 octal) on the UNIVAC 1108 with 36-bit words.

All of the programs are written in FORTRAN V and are system independent with the exception of subroutines SEARCH and WFILE. This subroutine is a tape positioning routine and should be checked for system compatibility.

The parameters listed in the flow charts are defined in Section 5.10.

The basic grid system used in this model, displayed in Fig. 4.2, is a 300 foot lattice. A driving tide is input for the open sea boundary. The boundary conditions at the open water boundaries in the three branches of the inlet are extrapolated as explained in Sections 2.2 and 3.1.

Input and output to the program are accomplished by both punched cards and tape. All tape input/output is by standrad FORTRAN binary reads and writes. That is, by "READ(N) List" and "WRITE(N) List" where N is the logical unit number and "List" is a variable list. Flow charts and listings of the programs are given in this section. There is a restart capability present in the program.

All parameters in subroutine calls must be specified.

5.2 Program MASON

Program MASON is the main program. It controls the overall flow of the Masonboro model. The input of data, initialization of parameters, the actual flow computations, and output of results are controlled by this routine. Intermediate results can be printed at any specified time interval. A flow chart of the program is given in Fig. 5.1 and a listing is given in Table 5.1

5.3 Subroutine FLAGS

This subroutine reads in the basic flag information and defines a flag code for use by program MASON. The flag field code describes the flow conditions (free flow, barrier, etc.) at the cell walls so that the correct computations can be performed. The flag field is dynamic; that is, the flag field changes as the water level changes to allow flooding and draining of cells. Details of the flag system are given in Section 3.2. The initial flag field is set up for a water level of 0.0 feet. It is later modified to the initial water level by subroutine MOD.

Subroutine FLAGS is called by the statement

```
CALL FLAGS (LHI, LHJ, LHK)
```

where LHI = a two-dimensional array of the I subscripts of the special tidal input cells,

LHJ = a two-dimensional array of the J subscripts of the special tidal input cells, and

LHK = a one-dimensional array containing the number of special tidal input cells of each type.

A flow chart of the program is given in Fig. 5.2 and a program listing is given in Table 5.2.

5.4 Subroutine MOD

This subroutine modifies the flag field when called according to newly computed water levels. That is, it checks if any given cell should be flooded or dry and changes the corresponding flag accordingly. To tell whether or not a cell should be flooded, the program checks the elevation of water level in the adjacent cells to see if it is greater than the elevation of the cell by some tolerance specified by the user in the data cards. If a cell is already flooded and the water level in an adjacent cell drops to within the specified amount of the bottom elevation then the cell is taken out of the computations and treated as dry land. In checking the cells the program scans from cell (I = 1, J = 1) to cell (I = IMAX, J = 1), increments J and repeats, i.e., it scans the grid by rows.

During flood tide subroutine MOD is called at a regular interval specified by the user (variable NSTEP). During ebb tide it is called whenever a negative depth is calculated. The calling statement is

```
CALL MOD ( TIME, IPRMOD, IOPP)
```

where TIME = the time of day in minutes,

IPRMOD = number of calls to MOD between calls to subroutine MODPRI by MOD, and

IOPP = the print control parameter for the call to MODPRI (see subroutine MODPRI, Section 5.8).

A flow chart for the program is given in Fig. 5.3 and a listing in Table 5.3.

5.5 Subroutine TRACE

This subroutine traces flooding backward through the grid. That is, if a cell is initially flooded through the left, top, or right wall, then it is possible that the cells to its left and bottom will be flooded by it. However, subroutine MOD has already scanned these cells and would not detect this. Therefore, whenever a cell is flooded through the left, top, or right wall, subroutine TRACE is called to trace back through the grid and flood any cells, if necessary.

The call to subroutine TRACE is

CALL TRACE (I, J, Z)

where I = the I subscript of the cell just flooded,

J = the J subscript of the cell just flooded, and

Z = the bottom elevation array.

A flow chart of the program is given in Fig. 5.4 and a listing in Table 5.4.

5.6 Subroutine WIND

This subroutine calculates the wind effect (wind stresses) as a function of time. The wind data is input via data cards. The data consists of the wind speed, wind direction, and the time, in minutes, at which the wind changes either speed, direction, or both. Also required is the heading of the positive x axis with respect to north. There is currently a capacity for ten (10) changes in the wind. In order to increase the number of changes, the dimension card must be modified. The wind changes are currently treated as step functions with no interpolation between

wind changes. That is, a given wind change applies until the next wind change occurs.

The calling sequence is

CALL WIND (TIME, XW, YW)

where TIME = the time,

XW = x component of the wind stress per unit mass in units of ft^2/min^2, and

YW = y component of the wind stress per unit mass in units of ft^2/min^2.

The values of XW and YW are returned from the subroutine. The first call to WIND inputs the wind data and the variables XW and XY are dummies.

Whenever the wind conditions change this subroutine prints out a message indicating this along with the new values of the components of wind stress.

A flow chart is given in Fig. 5.5 and a program listing is given in Table 5.5.

5.7 Subroutine TIDAL2

This subroutine inputs the tide data and computes the driving tide as a function of time. Input is via data cards. The calling sequence is

CALL TIDAL (TIME + DT2, TIDE)

where DT2 = one-half the time step, and

TIDE = the returned value of the tide height.

The first call to TIDAL inputs the tide data for half-hour intervals and no computations are performed. Each succeeding call calculates the present tide height using linear interpolation between the half-hour values. No additional data is required.

A flow chart is given in Fig. 5.6 and a program listing is given in Table 5.6.

5.8 Subroutine PRINT1

This subroutine handles the printing of output from the program. There are three forms that the printed output may take; 1) the water elevation, and the magnitude and direction of the cell-centered velocity at selected cells, 2) water level and velocity at the five stations, or 3) complete arrays of specified variables. This subroutine has three entry points: P1PREP, PRINT1, and MODPRI. The entry point MODPRI is independent of the other two. Also the water level and velocities at the five stations can be saved to be punched latter.

The entry points P1PREP and PRINT1 are interconnected. These two entry points control the print-out of the first form of output. The entry point P1PREP must be called first. The calling statement is "CALL PRINT1." This entry point is used only once and it directs the subroutine to read in the necessary control information.

Entry point PRINT1 has three functions: 1) printing of tidal elevations and magnitudes at selected cells whose indices are read from cards via entry point PIPREP, 2) printing a table of tidal elevations and velocities for the five stations at Masonboro Inlet, and 3) storing of tidal elevations and velocities at the five stations printed under 2 for punching when TLIM is reached. Which of the three done is controlled by the parameter IPCH read from data card. There are four possibilities

IPCH = 0 - information for specified cells and stations printed but nothing stored, e.g. 1 and 2 above done

= 1 - all of the above done

= 2 - specified cells not printed but station information is printed and stored, i.e. 2 and 3 above done

= 3 - specified cells printed and station information stored for punching, i.e. 1 and 3 above done.

The calling statement is

CALL PRINT1 (TIME, TIDE)

where TIME = the present time, and

TIDE = the present tide level.

The form of the output is

(TIME)(TIDE)(← water level with respect to datum level →)
(← magnitude of velocity vector (cfs) →)
(← direction of velocity vector with respect to positive x axis →)

The cells used to determine the information corresponding to the five stations for which data were provided are shown in Table 5.7. The velocities indicated are cell-centered velocities.

The entry point MODPRI is used for the second form of output. Arrays of any one or more of the following variables may be printed: 1) flag field, 2) water level, 3) water depth, 4) flow in x-direction, 5) flow in y-direction, 6) velocity in x-direction, and the velocity in the y-direction. The calling statement is

CALL MODPRI (IOPP, TIME)

where IOPP = parameter specifying variables to be printed, add up the numbers corresponding to the parameters to be printed listed in Table 5.8,

TIME = time of day in hours.

A flow chart is given in Fig. 5.7 and a program listing in Table 5.9.

5.9 Subroutine SEARCH

This is a tape positioning program using specially formatted tapes. It is used in conjunction with subroutine WFILE.

When called, SEARCH will position a tape immediately following the file number requested; that is, at the beginning of the first data block following the requested file number.

When writing tapes, the usual will be one of two possibilities: begin a new tape, or add on data to a tape with other files on it.

A. SEARCH will initialize a new tape if the file number is 1; and the tape will be ready, when control is returned, for writing of a data block.

B. When adding new data to a tape with files already on it, all files after the requested file are no longer accessible. Therefore, if one does not wish to destroy any of the files on the tape the

file that should be the last file with data in it plus one.

There are two entry points: SRCHOF and SRCHIF. SRCHOF is called prior to writing on tape, and SRCHIF is called prior to reading tape. The calling sequences are

CALL SRCHOF (FN, LUN)

and CALL SRCHIF (FN, LUN)

where FN is the file number and LUN is the FORTRAN logical unit number. File number zero cannot be requested.

The following error stops and diagnostics are present:

A. "CANNOT LOCATE REQUESTED FILE NUMBER. REQUESTED FILE NUMBER = X. LAST FILE NUMBER ON TAPE = Y OCTAL." ERROR ON LOGICAL UNIT Z."

This message will probably mean that the file number requested was larger than the null file number. The last file number printed is <u>not</u> the null file, but the last file on tape with data; that is, the number of the null file minus 1. All numbers are in octal. Job is aborted.

B. "FILE MARK NOT FOLLOWED BY FILE NUMBER. FILE TO BE LOCATED WAS X OCTAL. ERROR ON LOGICAL UNIT Z."

Job is aborted.

Two decks are provided: 1) UNIVAC exec 2 assembly language deck, and 2) the equivalent binary deck. A listing of the assembly language deck is given in Table 5.9.

5.10 Subroutine WFILE

WFILE is a program that writes file numbers and end-of-tape sentinals on magnetic tape. Specifically, it writes a file mark, a file number, another file mark, a 7777 77770000, and then backspaces the tape two blocks to just before the second file mark. This routine does no other tape positioning, and the tape must be properly positioned prior to the WFILE call.

This routine has two entry points: WFILE and WFILEF. The first -- without the "F" suffix -- is to be used for calls using alphabetic (field data) tape unit designations. The second is for calls using FORTRAN logical unit number designations. For FORTRAN calls the calling sequence is

```
CALL WFILEF (FN, LUN)
```

where FN = a constant or variable containing file number, and

LUN = a constant or variable containing logical unit number.

Two decks are provided, 1) the UNIVAC 1108 Exec 2 assembly language deck, and 2) the equivalent binary deck. A listing of the assembly language deck is given in Table 5.10.

TABLE 5.1

LISTING FOR PROGRAM MASON

```
      COMMON/A/QX(30,25),QY(30,25),D(30,25),IFLAG(30,25),H(30,25),IMAX,
     1     JMAX,DELTA,JB,JBR,IBU,PUNV(5,4,50),NPU,PCHT(50),IPCH
      COMMON /B/DUM(30,25),DX,DY
      DIMENSION QXN(30,25),QYN(30,25),HN(30,25),Z(30,25),F(30,25),
     1ZB(25),CS(25),DYB(25)
      DIMENSION LHK(3),LHI(3,25),LHJ(3,25)
      DIMENSION ITIT(3,5,4)
      DATA ITIT/'STATION 1 - NORTH','STATION 2 - NORTH','STATION 3 - NORT
     1TH','STATION 4 - WEST','STATION 5 - WEST','STATION 1 - CENTER','ST
     2ATION 2 -CENTER','STATION 3 - CENTER','STATION 4 - CENTER',
     3'STATION 5 - CENTER','STATION 1 - SOUTH','STATION 2 - SOUTH',
     3'STATION 3 - SOUTH','STATION 4 - EAST','STATION 5 - EAST',
     5'STATION 1 - TIDE','STATION 2 - TIDE','STATION 3 -TIDE',
     6'STATION 4 - TIDE','STATION 5 - TIDE'/
C                                                                       EST   7
C       ************************************************************   EST   8
C       * IMAX - NUMBER OF COLUMNS IN CELL STRUCTURE                *   EST   9
C       * JMAX - NUMBER OF ROWS IN CELL STRUCTURE                   *   EST  10
C       * NREEF - NUMBER OF REEFS (BARRIERS)                        *   EST  11
C       * IECHO = 0  NO ECHO PRINT OF INPUT DATA                    *   EST  14
C       *       = 1  ECHO PRINT INPUT DATA                          *   EST  15
C       *       = 2  ECHO PRINT INPUT DATA EXCEPT FRICTION COEF.    *   EST  16
C       * ITAPE - OUTPUT FILE NUMBER (=0 IF NO TAPING)              *   EST  17
C       * IQYQXH - INPUT FILE NUMBER (=0 IF NO INPUT STARTING COND.)*   EST  18
C       * INETFL - .GT. 0 OUTPUT FILE NUMBER FOR NET VELOCITIES     *   EST  20
C       *        - .EQ. (=) 0 NO OUTPUT OF NET VELOCITIES           *   EST  21
C       *        - .LT. 0 NET VELOCITIES PRINTED ONLY               *   EST  22
C       * DT - TIME INCREMENT (MINUTES)                             *   EST  23
C       * TMAX - LENGTH OF RUN (HOURS)                              *   EST  24
C       * PTIME - PRINT TIME INTERVAL (MINUTES)                     *   EST  25
C       * DS - CELL SIZE (FEET)                                     *   EST  26
C       * ZB - BARRIER ELEVATIONS W.R.T. MSL (FEET)                 *   EST  29
C       * DYB - BARRIER WIDTHS (FEET)                               *   EST  30
C       * OMEGA - ANGULAR VELOCITY OF THE EARTH (RAD/SEC)           *   EST  34
C       * PHI - LATITUDE (DEGREES)                                  *   EST  35
C       * Z - BOTTOM ELEVATION W.R.T. MSL (FEET)   INPUT AS         *   EST  36
C       *     POSITIVE BUT SIGN IS LATER CHANGED TO NEGATIVE        *   EST  37
C       * IFLAG - CELL IDENTIFICATION ARRAY                         *   EST  38
C       * F - FRICTION COEFFICIENTS                                 *   EST  39
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
C         ************************************************************  EST  46
C                                                                        EST  47
C      **************************************************************
C      STAGE 1.0 - SET CONSTANTS
C      **************************************************************
C                                                                        EST  49
1      FORMAT (1H1)                                                      EST  50
       G=32.1725*3600.0                                                  EST  51
       GP=0.142*32.1725                                                  EST  52
       TPER=1500.0                                                       EST  53
       PI=3.1416                                                         EST  54
       R=0
C                                                                        EST  55
C      **************************************************************
C      STAGE 2.0 - READ IN CONTROL PARAMETERS
C      **************************************************************
C                                                                        EST  57
       READ 2,IMAX,JMAX,NREEF,IECHO,ITAPE,IQYQXH,IPRT,
      1IPRMOD,NSTEP,JR,JBR,IBU,IOPP,IOPP1,IPCH
2      FORMAT (16I5)                                                     EST  60
       READ 3,DT,TMAX,PTIME,DS,TTAPE,VAL,TLIM
     3 FORMAT(7F10.3)
       READ 100,DELTA,DELTA2,TIME,TINT,TPRMB,TPRME
   100 FORMAT(6F10.3)
C
C      **************************************************************
C      STAGE 3.0 - READ IN REEF INFORMATION
C      **************************************************************
C
       IF (NREEF.EQ.0) GO TO 5                                           EST  66
       READ 4, (CS(I),ZB(I),DYB(I),I=1,NREEF)                            EST  67
4      FORMAT (3F10.0)                                                   EST  68
C                                                                        EST  71
C      **************************************************************
C      STAGE 4.0 - READ IN CORIOLIS INFORMATION
C      **************************************************************
C                                                                        EST  73
     5 READ 4,OMEGA,PHI
       PHI=PHI*PI/180.                                                   EST  75
       OMEGA=2.*OMEGA*SIN(PHI)                                           EST  76
C                                                                        EST  77
C      **************************************************************
C      STAGE 5.0 - SET CELL DIMENSIONS
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
C     ***********************************************************************
C                                                                             EST  79
      DX=DS                                                                   EST  80
      DY=DS                                                                   EST  81
C                                                                             EST  82
C     ***********************************************************************
C     STAGE 6.0 - READ SEA BED ELEVATIONS IN FEET W.R.T. DATUM
C                 BELOW DATUM POSITIVE--- ABOVE DATUM NEGITIVE
C     ***********************************************************************
C                                                                             EST  84
      DO 7 J=1,JMAX                                                           EST  85
7     READ 8, (Z(I,J),I=1,IMAX)                                               EST  86
8     FORMAT (20F4.0)                                                         EST  87
C                                                                             EST  88
C     ***********************************************************************
C     STAGE 7.0 - READ IN FRICTION FACTOR SCALLING, ECHO PRINT, AND
C                 COMPUTE FRICTION FACTORS
C     ***********************************************************************
C
      DO 110 J=1,JMAX
110   READ 111,(F(I,J),I=1,IMAX)
  111 FORMAT(11F7.3)
      DO 11 J=1,JMAX                                                          EST  97
      DO 11 I=1,IMAX                                                          EST  98
      IF(Z(I,J) .LT. 0. ) GO TO 10
      F(I,J)=SQRT(0.0025*F(I,J)*(2.0*Z(I,J))**0.333/(4.0*GP))
      GO TO 11
   10 F(I,J)=VAL*F(I,J)
11    CONTINUE                                                                EST 100
C                                                                             EST 101
C     ***********************************************************************
C     STAGE 8.0 - ECHO PRINT INPUT DATA
C     ***********************************************************************
C                                                                             EST 103
      PRINT 12                                                                EST 104
12    FORMAT (5X,'ECHO PRINT ALL INPUT DATA',/////)                           EST 105
      PRINT 13,IMAX,JMAX,NREEF,IECHO,IQYQXH,DT,DS,IPRMOD,NSTEP,DELTA,
     1DELTA2,JB,JBR,IBU,VAL,TINT,IPRT,PTIME,IOPP,IOPP1,ITAPE,TTAPE,
     2IPCH,TIME,TMAX,TLIM,TPRMB,TPRME
   13 FORMAT(5X,'ECHO PRINT OF CONTROL PARAMETERS'//
     15X,'IMAX   =',I5,/5X,'JMAX   =',I5/5X,'NREEF  =',I5/5X,'IECHO  =',
     2I5,/5X,'IQYQXH =',I5/5X,'DT     =',F11.5/5X,'DS     =',F8.2/5X,
     3'IPRMOD =',I5/5X,'NSTEP  =',I5/5X,'DELTA  =',F8.2/5X,'DELTA2 =',
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
     4F8.2/5X,'JB     =',I5/5X,'JBR    =',I5/5X,'IBU    =',I5/5X,
     5'VAL    =',F11.5/5X,'TINT   =',F8.2/5X,'IPRT   =',I5/5X,
     6'PTIME  =',F8.2/5X,'IOPP   =',I5/5X,'IOPP1  =',I5/5X,'ITAPE  =',I5
     7/5X,'TTAPE  =',F8.2/5X,'IPCH   =',I5/5X,'TIME   ='F8.2/5X,'TMAX
     8=',F8.2/5X,'TLIM   =',F8.2/5X,'TPRMB  =',F8.2/5X,'TPRME  =',F8.2/)
      IF ( IQYQXH .EQ. 0)TMAX=TMAX+TIME
      IF (IECHO.EQ.0) GO TO 31                                          EST 115
      PRINT 14                                                          EST 116
   14 FORMAT(5X,'REEFS,TIDES,BARRIER ELEV. AND DISCHARGE COEFFICIENTS',/
     1/)
      PRINT 15                                                          EST 119
15    FORMAT (5X,'     I    CS    ZB     DYB'//)                        EST 120
      PRINT 16, (I,CS(I),ZB(I),DYB(I),I=1,NREEF)                        EST 121
16    FORMAT (5X,I5,2F5.1,F10.1)                                        EST 122
      PRINT 19                                                          EST 127
19    FORMAT (//5X,'DATA FOR CORIOLIS FORCES',/)                        EST 128
      PRINT 20, OMEGA,PHI                                               EST 129
20    FORMAT (5X,'OMEGA = ',F10.3,//,5X,'PHI   = ',F10.3,//)            EST 130
      PRINT 1                                                           EST 131
      PRINT 21                                                          EST 132
21    FORMAT (5X,'LAND AND SEABED ELEVATIONS',//)                       EST 133
      PRINT 22,(J,J=1,JMAX)
   22 FORMAT(/,' I/J',I4,24I5,/)
      DO 23 I=1,IMAX                                                    EST 137
23    PRINT 24, I,(Z(I,J),J=1,JMAX)                                     EST 138
   24 FORMAT(I3,2X,25F5.1)
      PRINT 1                                                           EST 140
      IF (IECHO.EQ.2) GO TO 31                                          EST 148
      PRINT 28                                                          EST 149
28    FORMAT (5X,'FRICTION FACTORS',//)                                 EST 150
      DO 29 J=1,JMAX                                                    EST 151
29    PRINT 30, J,(F(I,J),I=1,IMAX)                                     EST 152
30    FORMAT (/10X,'J = ',I2/(10F8.5))                                  EST 153
31    CONTINUE                                                          EST 154
C                                                                       EST 155
C     ***************************************************************
C     STAGE 9.0 - SET INITIAL CONDITIONS AND OTHER CONSTANTS
C     ***************************************************************
C                                                                       EST 157
      TMAX=TMAX*60.-0.001
      TPRMB=TPRMB*60.
      TPRME=TPRME*60.
      TLIM=TLIM*60.
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
      TIME=TIME*60.
      TTEST=TIME+TTAPE
      PRISM=0.
      ST=0.
      IF ( TIME .LT. 0. ) ST=TIME
      HNA4=0.
      HA4=0.
      HA42=0.
      IPM=2
      ILE=0
      ICO=0
      IMAXM1=IMAX-1
      JMAXM1=JMAX-1
      TNET=TMAX-TPER+DT                                                 EST 169
      IF ( TNET .LT. 0. ) TNET=TMAX+DT
      TPRINT=0.0                                                        EST 170
      DT2=DT/2.0                                                        EST 171
      GPDT=GP*DT                                                        EST 172
      DTO2DX=DT/(2.0*DX)                                                EST 173
      DTO2DY=DT/(2.0*DY)                                                EST 174
      GDTO2X=G*DTO2DX                                                   EST 175
      GDTO2Y=G*DTO2DY                                                   EST 176
C
C     ****************************************************************
C     STAGE 10.0 - ZERO OUT ARRAYS
C     ****************************************************************
C
      DO 33 J=1,JMAX                                                    EST 180
      DO 33 I=1,IMAX                                                    EST 181
      QX(I,J)=0.0                                                       EST 182
      QY(I,J)=0.0                                                       EST 183
      QXN(I,J)=0.0                                                      EST 184
      QYN(I,J)=0.0                                                      EST 185
      H(I,J)=0.0                                                        EST 186
      HN(I,J)=0.0                                                       EST 187
      Z(I,J)=-Z(I,J)                                                    EST 194
      D(I,J)=H(I,J)-Z(I,J)                                              EST 195
33    CONTINUE                                                          EST 196
C
C     ****************************************************************
C     STAGE 11.0 - CHECK SOURCE OF INITIAL VALUES AND JUMP
C                  IQYQXH .LT. 0  = INITIAL VALUES FROM CARDS
C                  IQYQXH .EQ. 0  = INITIAL VALUES - H=0 AND FLOWS=0
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
C                IQYQXH .GT. 0 = INITIAL VALUES READ FROM TAPE
C                                NUMBER IS FILE NUMBER ON TAPE
C     ******************************************************************
C
      IF ( IQYQXH)36,36,34
C
C     ******************************************************************
C     STAGE 11.1 - READ INITIAL VALUES AND OTHER DATA FROM TAPE
C     ******************************************************************
C
34    CALL SRCHIF (IQYQXH,1)                                            EST 199
      READ(1)TIME,LHK,LHI,LHJ
      READ(1)PRISM,H1,H2,H3,H4,CON1,CON2,CON3,CON4,IPM,HNA4,HA4,HA42
     1,NPU,PUNV,PCHT
      READ(1) IFLAG
      READ (1) H                                                        EST 200
      READ (1) QX                                                       EST 201
      READ (1) QY                                                       EST 202
C
C     ******************************************************************
C     STAGE 11.2 - IF INITIAL TIME TO BE DIFFERENT FROM THAT ON TAPE
C                  SET INITIAL TIME
C     ******************************************************************
C
      IF ( ST .LT. 0. ) TIME=-ST
      TMAX=TMAX+TIME
      PRINT 183,TIME,TMAX
C
C     ******************************************************************
C     STAGE 11.3 - ECHO PRINT CONTROL PARAMETERS READ FROM TAPE
C     ******************************************************************
C
      PRINT 8885,H1,H2,H3,H4,CON1,CON2,CON3,CON4,IPM,PRISM
 8885 FORMAT(8E12.5,I5,E12.5)
  183 FORMAT(5X,'RESTARTED FROM TAPE AT TIME =',E14.7/5X,'REVISED TMAX='
     *,F12.5)
      TTEST = TIME + TTAPE
      GO TO 39                                                          EST 203
C
C     ******************************************************************
C     STAGE 11.4 - END OF TAPE INPUT
C     ******************************************************************
C
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
   36 CALL FLAGS(LHI,LHJ,LHK)
C
C     *****************************************************************
C     STAGE 11.5 - TEST IF INITIAL DATA IS ON CARDS
C     *****************************************************************
C
      IF ( IQYQXH .EQ. 0 ) GO TO 41
C
C     *****************************************************************
C     STAGE 11.6 - INITIAL DATA ON CARDS
C     *****************************************************************
C
35    READ 37,H                                                         EST 204
37    FORMAT(8E10.4)                                                    EST 205
      READ 38,QX                                                        EST 206
      READ 38, QY                                                       EST 208
38    FORMAT(8E10.4)                                                    EST 208
39    CONTINUE                                                          EST 209
C
C     *****************************************************************
C     STAGE 12.0 - INITIAL VALUES READ FROM TAPE OR CARDS, THEREFORE
C                  NEED TO CALCULATE DEPTH ARRAY
C     *****************************************************************
C
      DO 40 J=1,JMAX                                                    EST 211
      DO 40 I=1,IMAX                                                    EST 212
40    D(I,J)=H(I,J)-Z(I,J)                                              EST 213
      IF ( ST .GE. 0. ) GO TO 45
      GO TO 46
C
C     *****************************************************************
C     STAGE 13.0 - SET DEPTHS AND WATER LEVELS IN DRY CELLS TO -1000
C     *****************************************************************
C
   41 DO 44 J=1,JMAX
      DO 44 I=1,IMAX
      IF ( IFLAG(I,J) .GE. 237 .AND. IFLAG(I,J) ,LT. 256 ) GO TO 43
      IF ( D(I,J) .EQ. -1000. ) GO TO 44
      H(I,J)=TINT
      D(I,J)=TINT-Z(I,J)
   43 IF ( D(I,J) .GT. .00000002 ) GO TO 44
      D(I,J)=-1000.
      H(I,J)=-1000.
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
   44 CONTINUE
C
C     ******************************************************************
C     STAGE 14.0 - CALL MOD TO SET FLAG FIELD FOR INITIAL WATER LEVEL
C     ******************************************************************
C
   46 CALL MOD(TIME,IPRMOD,Z,19)
C
C     STAGE 15.0 - CALL SURROUTINES TO INITIALIZE THEM
C     ******************************************************************
C
   45 CALL WIND(TIME,XW,YW)
      CALL TIDAL2(TIME+DT2,TIDE)
      CALL P1PREP                                                       EST 216
      CALL PRINT1(TIME,TIDE)
C                                                                       EST 218
C
C     ******************************************************************
C     STAGE 16.0 - BEGIN COMPUTIONS - CONTROL CANNOT RETURN TO ANY
C                  POINT ABOVE THIS POINT FROM BELOW
C     ******************************************************************
C
C                                                                       EST 220
42    CALL WIND (TIME,XW,YW)                                            EST 221
C
C     ******************************************************************
C     STAGE 17.0 - INCREMENT OR INITIALIZE COUNTERS AND TIME INDICATORS
C     ******************************************************************
C
      ICO=ICO+1
      LOOP=0
      TIME=TIME+DT                                                      EST 223
      TPRINT=TPRINT+DT                                                  EST 224
C
C     ******************************************************************
C     STAGE 18.0 - FIND NEW TIDES AT BOUNDARY
C     ******************************************************************
C
      CALL TIDAL2(TIME+DT2,TIDE)
  420 KB=0
      KD=0                                                              EST 227
      KT=0                                                              EST 228
      KF=0
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
      IDUP=1
C
C     ******************************************************************
C     STAGE 19.0 - THE LOOP TO 518 PERFORMS ALL COMPUTATIONS BUT THOSE
C                  FOR THE SPECIAL BOUNDARY CONDITIONS
C     ******************************************************************
C
      DO 518 J=1,JMAX
      DO 518 I=1,IMAX
      IP1=I+1
      IM1=I-1
      JP1=J+1
      JM1=J-1
      IFL=IFLAG(I,J)
      IHO=IFLH
      IX=0
  499 IFLH=FLD(33,3,IFL)+1
C     ******************************************************************
C     STAGE 19.1 - CHECK IF CELL IS ALWAYS DRY -- IF SO JUMP
C     ******************************************************************
C
      IF ( IFL .EQ. 502 ) GO TO 522
C
C     ******************************************************************
C    STAGE 19.2 - IF I=1 JUMP TO END OF LOOP
C     ******************************************************************
C
      IF ( I .EQ. 1 ) GO TO 518
      IFLV=FLD(30,3,IFL)+1
C
C     ******************************************************************
C     STAGE 19.3 - CODE FROM HERE TO 'END A' CALCULATES X-COMPONENT
C                  OF THE FLOWS
C     ******************************************************************
C
      GO TO (500,501,502,507,503,506,508,530),IFLH
  500 CONTINUE
      IF ( IFL .LT. 256 ) GO TO 700
      IF ((D(I,J)+D(IP1,J)) .GT. 0. ) GO TO 8867
      CALL MODPRI(127,TIME)
      PRINT 7776
 7776 FORMAT('   MODPRI CALL AFTER STATEMENT NO. 500')
      PRINT 8866,I,J,TIME
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
 8866 FORMAT(2I10,F12.5)
 8867 CONTINUE
C
C     ****************************************************************
C     STAGE 19.3.1 - CALCULATE TERMS FOR NON TIDAL FLAT FREE FLOW
C     ****************************************************************
C
      G1=1.0+GPDT*(F(I,J)+F(IP1,J))**2*SQRT(16.0*QX(I,J)*QX(I,J)+(QY(I,J
     1M1)+QY(IP1,JM1)+QY(I,J)+QY(IP1,J))**2)/(D(I,J)+D(IP1,J))**2.33
      GO TO 505
C
C     ****************************************************************
C     STAGE 19.3.2 - CALCULATE TERMS FOR TIDAL FLAT FREE FLOW
C     ****************************************************************
C
  700 G1=1.0+F(I,J)*DT*SQRT(16.0*QX(I,J)*QX(I,J)+(QY(I,JM1)+QY(IP1,JM1)+
     *QY(I,J)+QY(IP1,J))**2)/(D(I,J)+D(IP1,J))**2
      GO TO 505
C
C     ****************************************************************
C     STAGE 19.3.3 - CALCULATE TERMS FOR FLOW OVER REEF
C     ****************************************************************
C

  501 KB=KB+1
      DB=0.5*(H(IP1,J)+H(I,J))-ZB(KB)
      IF ( DB .LT. DELTA2 ) GO TO 506
      G1=1.0+DTO2DX*(D(IP1,J)+D(I,J))*ABS(QX(I,J))/(CS(KB)*DB)**2
C
C     ****************************************************************
C     STAGE 19.3.4 - CALCULATE X-COMPONENT FLOWS FOR FREE FLOW OR FLOW
C                    OVER REEF
C     ****************************************************************
C
  505 QXN(I,J)=(QX(I,J)+GDTO2X*(D(IP1,J)+D(I,J))*(H(I,J)-H(IP1,J))+(XW+O
     1MEGA*QY(I,J))*DT)/G1
      GO TO 507
C
C     ****************************************************************
C     STAGE 19.3.5 - TIDAL INPUT ON RIGHT WALL
C     ****************************************************************
C
  502 HN(IP1,J)=TIDE
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
      G1=1.0+GPDT*(F(I,J)+F(IP1,J))**2*SQRT(16.0*QX(I,J)*QX(I,J)+ (
     12.*(QY(I,JM1)+QY(I,J)))**2)/(D(I,J)+D(IP1,J))**2.33
      QXN(I,J)=(QX(I,J)+GDTO2X*(D(I,J)+D(IP1,J))*(H(I,J)-HN(I+1,J))+
     1XW*DT)/G1
      GO TO 507
C
C     ******************************************************************
C     STAGE 19.3.6 - TIDAL INPUT ON LEFT WALL
C     ******************************************************************
C
  503 HN(I,J)=TIDE
      G1=1.0+GPDT*(F(I,J)+F(IP1,J))**2*SQRT(16.0*QX(I,J)*QX(I,J)+
     1(2.*(QY(IP1,JM1)+QY(IP1,J)))**2)/(D(I,J)+D(IP1,J))**2.33
      QXN(I,J)=(QX(I,J)+GDTO2X*(D(I,J)+D(IP1,J))*(HN(I,J)-H(I+1,J))+
     1XW*DT)/G1
      GO TO 518
C
C     ******************************************************************
C     STAGE 19.3.7 - PERMANENT BARRIER
C     ******************************************************************
C
  530 IF ( IFL .EQ. 511 ) GO TO 521
      QXN(I,J)=0.
      GO TO 507
C
C     ******************************************************************
C     STAGE 19.3.8 - TEMPORARY BARRIER
C     ******************************************************************
C
  506 QXN(I,J)=0.
      IF ( IHO .LT. 5 ) GO TO 507
  508 IX=1
C
C     ******************************************************************
C
C     END A  ---  END OF STAGE 19.3  ---  END A

C
C     ******************************************************************
C     ******************************************************************
C
C     STAGE 19.4 - CODE FROM HERE TO 'END B' CALCULATES Y COMPONENT
C                  OF FLOW
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
C
C     *******************************************************************
C
  507 GO TO (510,511,512,518,513,516,519,531),IFLV
  510 CONTINUE
      IF ( (D(I,J)+D(I,JP1)) .GE. 0. ) GO TO 523
      PRINT 7775
 7775 FORMAT('    MODPRI CALLED AFTER STATEMENT NO. 510')
      CALL MODPRI(127,TIME)
  523 IF ( IFL .LT. 256 ) GO TO 710
C
C     *******************************************************************
C     STAGE 19.4.1 - CALCULATE TERMS FOR NON TIDAL FLAT FREE FLOW
C     *******************************************************************
C
      G2=1.0+GPDT*(F(I,J)+F(I,JP1))**2*SQRT(16.0*QY(I,J)*QY(I,J)+(QX(IM1
     1,J)+QX(I,J)+QX(IM1,JP1)+QX(I,JP1))**2)/(D(I,J)+D(I,JP1))**2.33
      GO TO 515
C
C     *******************************************************************
C     STAGE 19.4.2 - CALCULATE TERMS FOR TIDAL FLAT FREE FLOW
C     *******************************************************************
C
  710 G2=1.0+F(I,J)*DT*SQRT(16.*QY(I,J)*QY(I,J)+(QX(IM1,J)+QX(IM1,JP1)+
     *QX(I,J)+QX(I,JP1))**2)/(D(I,J)+D(I,JP1))**2
      GO TO 515
C
C     *******************************************************************
C     STAGE 19.4.3 - CALCULATE TERMS FOR FLOW OVER REEF
C     *******************************************************************
C
  511 KB=KB+1
      DB=0.5*(H(I,JP1)+H(I,J))-ZB(KB)
      IF ( DB .LT. DELTA2 ) GO TO 516
      G2=1.0+DTO2DY*(D(I,JP1)+D(I,J))*ABS(QY(I,J))/(CS(KB)*DB)**2
C
C     *******************************************************************
C     STAGE 19.4.4 - CALCULATE Y-COMPONENT OF FLOW
C     *******************************************************************
C
  515 QYN(I,J)=(QY(I,J)+GDTO2Y*(D(I,JP1)+D(I,J))*(H(I,J)-H(I,JP1))+(YW-O
     1MEGA*QX(I,J))*DT)/G2
      GO TO 517
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
C
C     ******************************************************************
C     STAGE 19.4.5 - TIDAL INPUT ON TOP WALL
C     ******************************************************************
C
  512 HN(I,JP1)=TIDE
      G2=1.0+GPDT*(F(I,J)+F(I,JP1))**2*SQRT(16.0*QY(I,J)*QY(I,J)+(2.*
     1(QX(I,J)+QX(IM1,J)))**2)/(D(I,J)+D(I,JP1))**2.33
      QYN(I,J)=(QY(I,J)+GDTO2Y*(D(I,J)+D(I,JP1))*(H(I,J)-HN(I,JP1))+
     1YW*DT)/G2
      GO TO 517
C
C     ******************************************************************
C     STAGE 19.4.6 - TIDAL INPUT ON BOTTOM WALL
C     ******************************************************************
C
  513 HN(I,J)=TIDE
      G2=1.0+GPDT*(F(I,J)+F(I,JP1))**2*SQRT(16.0*QY(I,J)*QY(I,J)+(2.*
     1(QX(I,JP1)+QX(IM1,JP1)))**2)/(D(I,J)+D(I,JP1))**2.33
      QYN(I,J)=(QY(I,J)+GDTO2Y*(D(I,J)+D(I,JP1))*(HN(I,J)-H(I,JP1))+
     1YW*DT)/G2
      GO TO 518
C
C     ******************************************************************
C     STAGE 19.4.7 - PERMANENT BARRIER
C     ******************************************************************
C
  531 QYN(I,J)=0.
      GO TO 517
C
C     ******************************************************************
C     STAGE 19.4.8 - TEMPORARY BARRIER
C     ******************************************************************
C
  516 QYN(I,J)=0.
      IF ( FLD(30,3,IFLAG(I,JM1)) .LT. 5 ) GO TO 517
  519 IX=IX+1
      IF ( IX .EQ. 2 ) GO TO 521
      IF ( IFLH .EQ. 4 ) GO TO 518
      IF ( H(I,J) .LT.-999.9 ) GO TO 521
  517 IF ( IHO .EQ. 4 ) GO TO 518
C
C     ******************************************************************
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
C
C     END B  ---  END OF STAGE 19.4  ---  END B
C
C     *****************************************************************
C     *****************************************************************
C
C     STAGE 19.5 - CALCULATE WATER LEVEL
C
C     *****************************************************************
C
      HN(I,J)=H(I,J)+DT*((QXN(IM1,J)-QXN(I,J))/DX+(QYN(I,JM1)-QYN(I,J))/
     1DY+R)
      GO TO 518
  522 IF ( IHO .EQ. 3 ) GO TO 518
C
C     *****************************************************************
C    STAGE 20.0 - THE LOOP TO 520 CALCULATES SPECIAL BOUNDARY CONDITIONS
C                 ON LEFT WALL
C     *****************************************************************
C
  521 HN(I,J)=-10000.
  518 CONTINUE
      IDEX=LHK(1)
      DO 520 JJ=1,IDEX
      I=LHI(1,JJ)
       J=LHJ(1,JJ)
      I1=I+1
      HN(I1,J)=-1000.
      IF ( FLD(29,1,IFLAG(I1,J)) .EQ. 1 ) GO TO 520
      DH1=HN(I+3,J)-H(I+3,J)
      DH2=HN(I+2,J)-H(I+2,J)
      HN(I1,J)=H(I1,J)+DH2+DH2-DH1
      IF ( J .LT. 15 ) GO TO 301
      IF ( DH2 .GT. 0. .AND. DH1 .GT. 0. ) GO TO 306
      IF ( HN(I1,J) .GT. H(I1,J)) HN(I1,J)=H(I1,J)-.001
      IF ( HN(I1,J) .LT. HN(I+2,J)) HN(I1,J)=HN(I+2,J)+.001
      GO TO 301
  306 IF ( HN(I1,J) .LT. H(I1,J)) HN(I1,J)=H(I1,J)+.001
      IF ( HN(I1,J) .GT. HN(I+2,J)) HN(I1,J)=HN(I+2,J)-.001
  301 QXN(I,J)= QXN(I1,J)-QYN(I1,J-1)+QYN(I1,J)+DX*(HN(I1,J)-H(I1,J))/DT
  520 CONTINUE
      IDEX=LHK(2)
C
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
C     ******************************************************************
C     STAGE 21.0 - THE LOOP TO 504 CALCULATE SPECIAL BOUNDARY CONITIONS
C                  ON THE RIGHT WALL
C     ******************************************************************
C
      DO 504 JJ=1,IDEX
      I=LHI(2,JJ)
      J=LHJ(2,JJ)
      HN(I,J)=-1000.
      IF ( FLD (33,3,IFLAG(I,J)) .GE. 5 ) GO TO 504
      IF ( IFLAG(I,J) .NE. 27 ) GO TO 730
      IF ( IFLAG(I,J-1) .LT. 256 ) GO TO 307
      HN(I,J)=HN(I,J-1)
      GO TO 302
  307 DH1=HN(I,J-2)-H(I,J-2)
      DH2=HN(I,J-1)-H(I,J-1)
      HN(I,J)=H(I,J)+DH2+DH2-DH1
      GO TO 302
  730 DH1=HN(I-2,J)-H(I-2,J)
      DH2=HN(I-1,J)-H(I-1,J)
      HN(I,J)=H(I,J)+DH2+DH2-DH1
      IF ( J .LT. 15 ) GO TO 302
      IF ( DH2 .GT. 0. .AND. DH1 .GT. 0. ) GO TO 305
      IF ( HN(I,J) .GT. H(I,J) ) HN(I,J)=H(I,J)-.001
      IF ( HN(I,J) .LT. HN(I-1,J) ) HN(I,J)=HN(I-1,J)+.001
      IF ( HN(I,J) .LT. HN(I,J-1) ) HN(I,J)=HN(I,J-1)
      GO TO 302
  305 IF ( HN(I,J) .LT. H(I,J) ) HN(I,J)=H(I,J)+.001
      IF ( HN(I,J) .GT. HN(I-1,J) ) HN(I,J) =HN(I-1,J)-.001
  302 QXN(I,J)=QXN(I-1,J)+QYN(I,J-1)-QYN(I,J)-DX*(HN(I,J)-H(I,J))/DT
  504 CONTINUE
      IDEX=LHK(3)
C
C     ******************************************************************
C     STAGE 22.0 - THE LOOP TO 514 CALCULATES SPECIAL BOUNDARY CONDITIONS
C                  ON TOP WALL
C     ******************************************************************
C
      DO 514 JJ=1,IDEX
      I=LHI(3,JJ)
      J=LHJ(3,JJ)
      IF ( IFLAG(I,J) .EQ. 27 ) GO TO 303
      HN(I,J)=-1000.
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
      IF ( FLD(30,3,IFLAG(I,J)) .GE. 5 ) GO TO 514
      DH1=HN(I,J-2)-H(I,J-2)
      DH2=HN(I,J-1)-H(I,J-1)
      HN(I,J)=H(I,J)+DH2+DH2-DH1
      IF ( DH2 .GT. 0. .AND .DH1 .GT. 0. ) GO TO 304
      IF ( HN(I,J) .GT. H(I,J) ) HN(I,J)=H(I,J)-.001
      IF ( HN(I,J) .LT. HN(I,J-1) ) HN(I,J)=HN(I,J-1)+.001
      GO TO 303
  304 IF (HN(I,J) .LT. H(I,J) )HN(I,J)=H(I,J)+.001
      IF ( HN(I,J) .GT. HN(I,J-1) ) HN(I,J) =HN(I,J-1)-.001
  303 QYN(I,J)=QYN(I,J-1)+QXN(I-1,J)-QXN(I,J)-DX*(HN(I,J)-H(I,J))/DT
  514 CONTINUE
C
C     ****************************************************************
C     STAGE 23.0 - THE CODE TO 'END C' CALCULATES THE TIDAL PRISM
C     ****************************************************************
C
      IF ( TIME .LT. TPRMB ) GO TO 610
      HA42=HA4
      HA4=HNA4
      IF ( IPM .LT. 2 ) GO TO 615
      IPM=1
      HA4=(H(14,15)+H(14,16)+H(15,15)+H(15,16))*.25
  615 HNA4=(HN(14,15)+H(14,16)+HN(15,15)+H(15,16))*.25
      IF ( IPM .EQ. 0 .AND. TIME  .LT. TPRME ) GO TO 600
      IF ( HNA4 .LT. HA4 .OR.  HA4 .LT. HA42 ) GO TO 610
      IF ( IPM .EQ. 0 ) GO TO 600
      IPM=0
      CON1=DX*DT
      H1=H(13,16)
      H2=H(14,16)
      H3=H(15,16)
      H4=H(16,16)
      CON2=CON1/(H2-Z(14,16))
      CON3=CON1/(H3-Z(15,16))
      CON4=CON1/(H4-Z(16,16))
      CON1=CON1/(H1-Z(13,16))
      PRINT 8888,H1,H2,H3,H4,CON1,CON2,CON3,CON4,DX,DT
 8888 FORMAT(10E12.5)
  600 PRISM=PRISM+CON1*QYN(13,16)*(HN(13,16)-H1)+CON2*QYN(14,16)*(HN(14,
     *16)-H2)+CON3*QYN(15,16)*(HN(15,16)-H3)+CON4*QYN(16,16)*(HN(16,16)-
     *H4)
C
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
C     ******************************************************************
C
C     END C  ---  END OF STAGE 23.0  ---  END C
C
C     ******************************************************************
C     ******************************************************************
C
C     STAGE 24.0 - CHECK IF FLOOD OR EBB TIDE -- IF FLOOD JUMP
C
C     ******************************************************************
C
  610 IF ( HN(11,12) .GT. H(11,12) ) GO TO 611
C
C     ******************************************************************
C     STAGE 25.0 - CHECK FOR CELL DRING UP
C     ******************************************************************
C
      KF=1
      DO 605 J=1,JMAX
      DO 605 I=2,IMAX
      IF  ( IFLAG(I,J) .GE. 256 ) GO TO 605
      DIFD=HN(I,J)-Z(I,J)
      IF ( DIFD .LT. -5.0 ) GO TO 605
      IF  ( DIFD .LT. 0. ) GO TO 606
  605 CONTINUE
      GO TO 611
  606 ICO=1
      IF ( LOOP .EQ. 1 ) GO TO 607
      LOOP=1
      PRINT 7778
 7778 FORMAT('    MOD CALL IN LOOP CHECK')
C
C     ******************************************************************
C     STAGE 25.1 - NEGITIVE DEPTH FOUND - CALL MOD AND RE PREFORM
C                  CALCULATIONS
C     ******************************************************************
C
      CALL MOD(TIME,IPRMOD,Z,IOPP)
      GO TO 420
  607 CALL MODPRI(19,TIME)
      PRINT 8877
 8877 FORMAT(1H1,15(/),'  ******************************************'/
     *'   HUNG IN LOOP'/'   CHECK ABOVE PRINTOUTS FOR NEGITIVE OR'/
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
     *'   ZERO DEPTHS AND CORRESPONDING FLARS'/
     *'   CAN BE RESTARTED FROM LAST TAPE DUMP ONCE ERROR CORRECTED'/
     *'  ************************************************************')
      PRINT 8844,I,J,TIME,DIFD,DELTA
 8844 FORMAT(5X,2I5,3E10.3)
      STOP
C
C     **************************************************************
C     STAGE 26.0 - REPLACE OLD VALUES WITH NEW
C     **************************************************************
C
  611 DO 74 J=1,JMAX
      DO 74 I=1,IMAX                                                    EST 344
      QX(I,J)=QXN(I,J)                                                  EST 345
      QY(I,J)=QYN(I,J)                                                  EST 346
      H(I,J)=HN(I,J)                                                    EST 347
      D(I,J)=H(I,J)-Z(I,J)                                              EST 348
74    CONTINUE                                                          EST 349
C
C     **************************************************************
C     STAGE 27.0 - CHECK IF MOD IS TO BE CALLED
C     **************************************************************
C
      IF ( KF .EQ. 1 ) GO TO 75
      IF(ICO .LT. NSTEP ) GO TO 75
      PRINT 7779
 7779 FORMAT('   MOD CALL AT REGULAR PLACE')
      CALL MOD(TIME,IPRMOD,Z,IOPP)
      ICO=0
C
C     **************************************************************
C     STAGE 28.0 - CHECK FOR PRINT
C     **************************************************************
C
   75 IF ( TPRINT .LE. PTIME ) GO TO 81
      TPRINT=0.0                                                        EST 374
      IF ( IPRT .EQ. 2 ) GO TO 80
      PRINT 7780
 7780 FORMAT('   MODPRI CALLED AT REGULAR PLACE')
      CALL MODPRI(IOPP1,TIME)
      IF ( IPRT .EQ. 1 ) GO TO 81
   80 CALL PRINT1(TIME,TIDE)
81    CONTINUE                                                          EST 376
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
C
C     ******************************************************************
C     STAGE 29.0 - CHECK FOR OUTPUT TO TAPE
C     ******************************************************************
C
      IF( ITAPE .EQ. 0 ) GO TO 90
      IF( TTEST-TIME ) ,,90
      CALL SRCHOF(ITAPE,1)
      WRITE(1)TIME,LHK,LHI,LHJ
      WRITE(1)PRISM,H1,H2,H3,H4,CON1,CON2,CON3,CON4,IPM,HNA4,HA4,HA42
     1,NPU,PUNV,PCHT
      WRITE(1) IFLAG
      WRITE(1) H
      WRITE(1) QX
      WRITE(1) QY
      CALL WFILEF(ITAPE+1,1)
      PRINT 319, ITAPE, TIME
  319 FORMAT(1X,'OUTPUT FILE',I5,' TO TAPE AT TIME =',E14.7)
      ITAPE = ITAPE + 1
      TTEST = TIME + TTAPE
      IDUP=0
      IF ( TPRINT .NE. 0 . ) CALL MODPRI(29,TIME)
   90 CONTINUE
C
C     ******************************************************************
C     STAGE 30.0 - CHECK IF TLIM HAS BEEN REACHED - IF SO JUMP
C     ******************************************************************
C
      IF ( TIME .GE. TLIM ) GO TO 82
C
C     ******************************************************************
C     STAGE 310 - CHECK IF TMAX HAS BEEN REACHED IF NOT LOOP
C     ******************************************************************
C
      IF (TIME.LT.TMAX) GO TO 42                                        EST 377
82    CONTINUE                                                          EST 380
C
C     ******************************************************************
C     STAGE 32.0 - CHECK FOR DUMP TO TAPE OF FINAL VALUES
C     ******************************************************************
C
      IF (ITAPE.EQ.0) GO TO 83                                          EST 381
      IF ( IDUP .EQ. 0 ) GO TO 83
```

TABLE 5.1 - continued

LISTING FOR PROGRAM MASON

```
      CALL SRCHOF (ITAPE,1)                                             EST 382
      WRITE(1)TIME,LHK,LHI,LHJ
      WRITE(1)PRISM,H1,H2,H3,H4,CON1,CON2,CON3,CON4,IPM,HNA4,HA4,HA42
     1,NPU,PUNV,PCHT
      WRITE(1) IFLAG
      WRITE (1) H                                                       EST 383
      WRITE (1) QX                                                      EST 384
      WRITE (1) QY                                                      EST 385
      CALL WFILEF (ITAPE+1,1)                                           EST 386
      PRINT 7781
 7781 FORMAT('     MODPRI CALLED AT TAPE WRITE')
      CALL MODPRI(29,TIME)
      CALL PRINT1(TIME,TIDE)
      PRINT 319,ITAPE,TIME
   83 IF ( TIME .LT. TLIM ) GO TO 88
      PRINT 91,PRISM
C
C     ******************************************************************
C     STAGE 33.0 - CHECK FOR PUNCHING OF TIDES AND VELOCITIES
C     ******************************************************************
C
   87 IF ( IPCH .EQ. 0 ) GO TO 88
      DO 84 I=1,5
      DO 84 J=1,4
      PUNCH 86,NPU,(ITIT(K,I,J),K=1,3)
      PUNCH 85,(PUNV(I,J,K),PCHT(K),K=1,NPU)
   84 CONTINUE
   88 CONTINUE
   85 FORMAT(16F5.2)
   86 FORMAT(I5,' 24.0',5X,3A6)
   91 FORMAT(1H1///10X,'TIDAL PRISM =',E12.5,' CUBIC FEET')
      END                                                               EST 399-
```

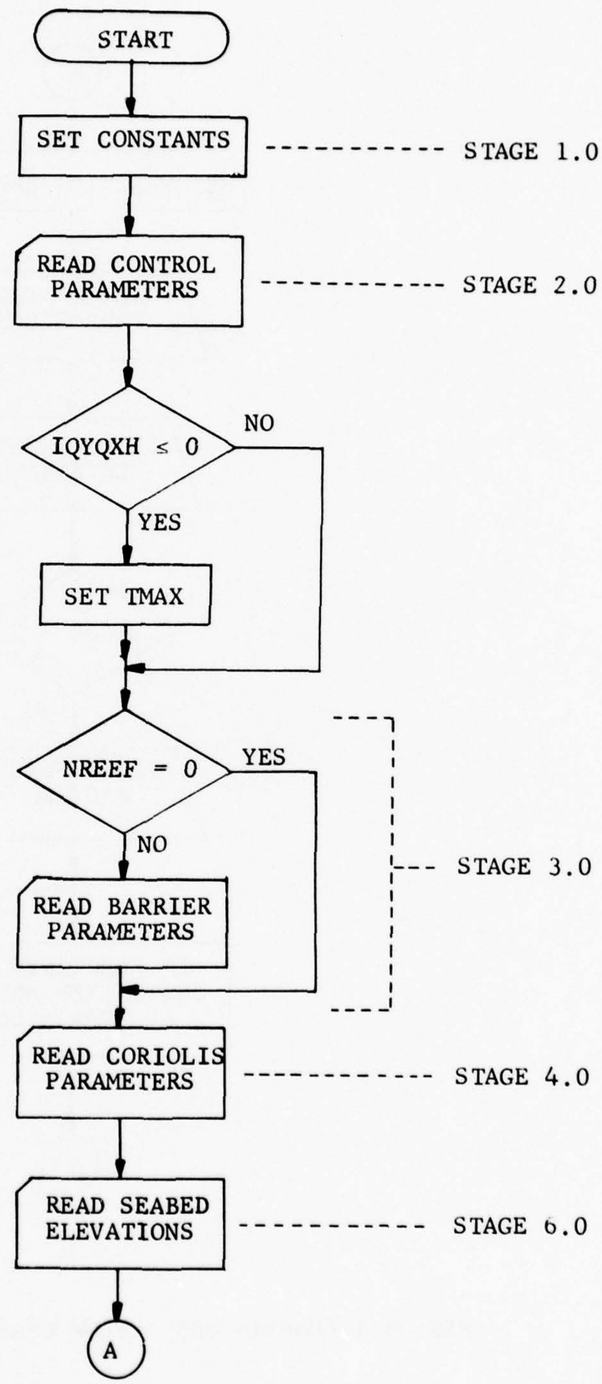


FIG. 5.1 FLOW CHART FOR PROGRAM MASON

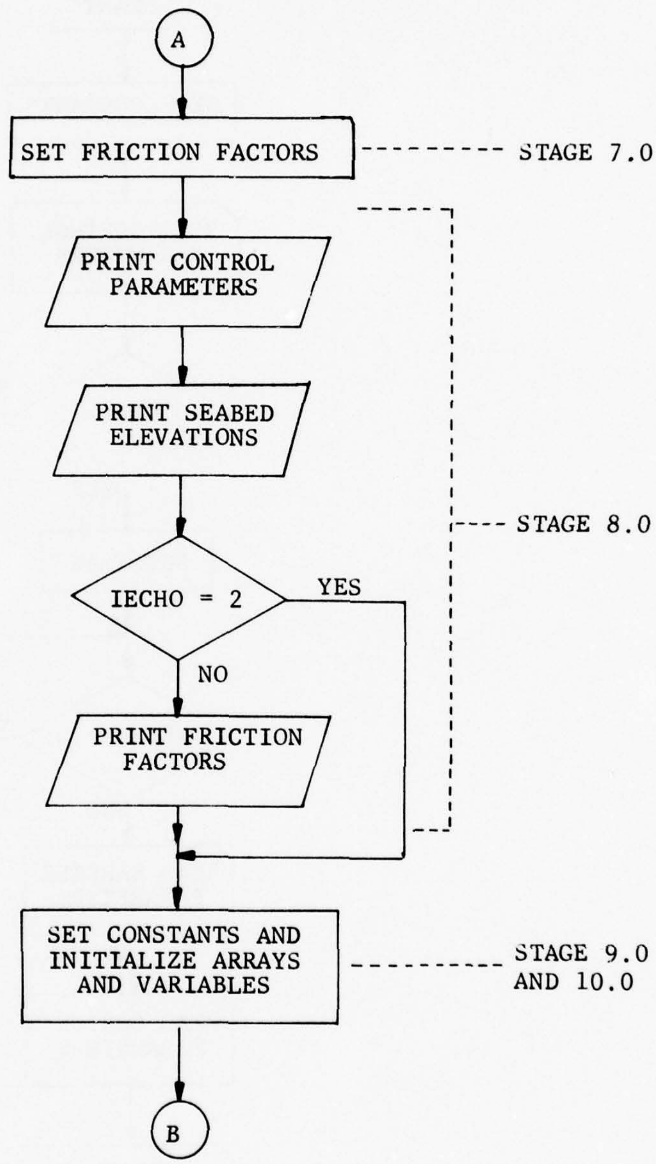


FIG. 5.1 (Continued) FLOW CHART FOR PROGRAM MASON

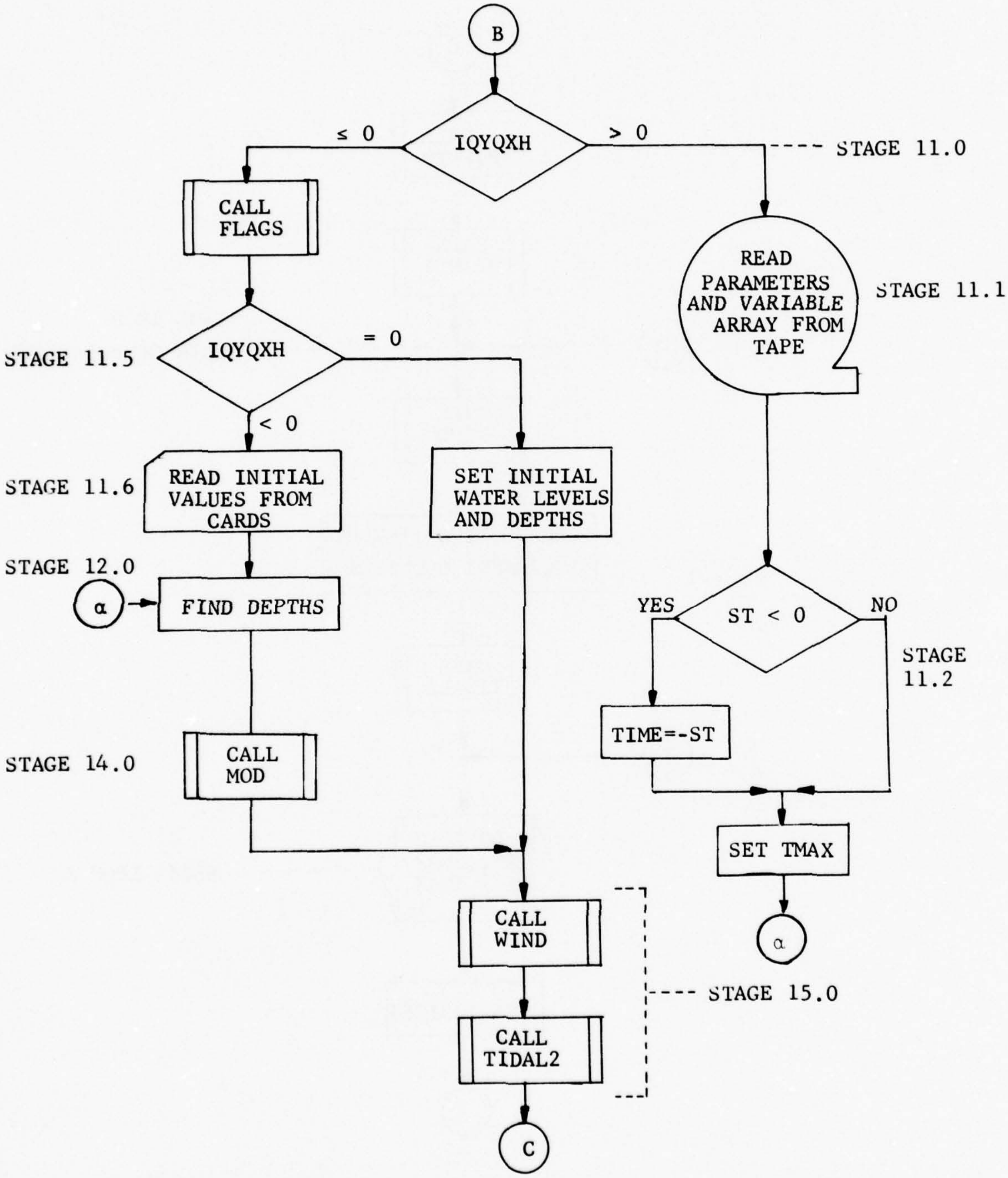


FIG. 5.1 (Continued) FLOW CHART FOR PROGRAM MASON

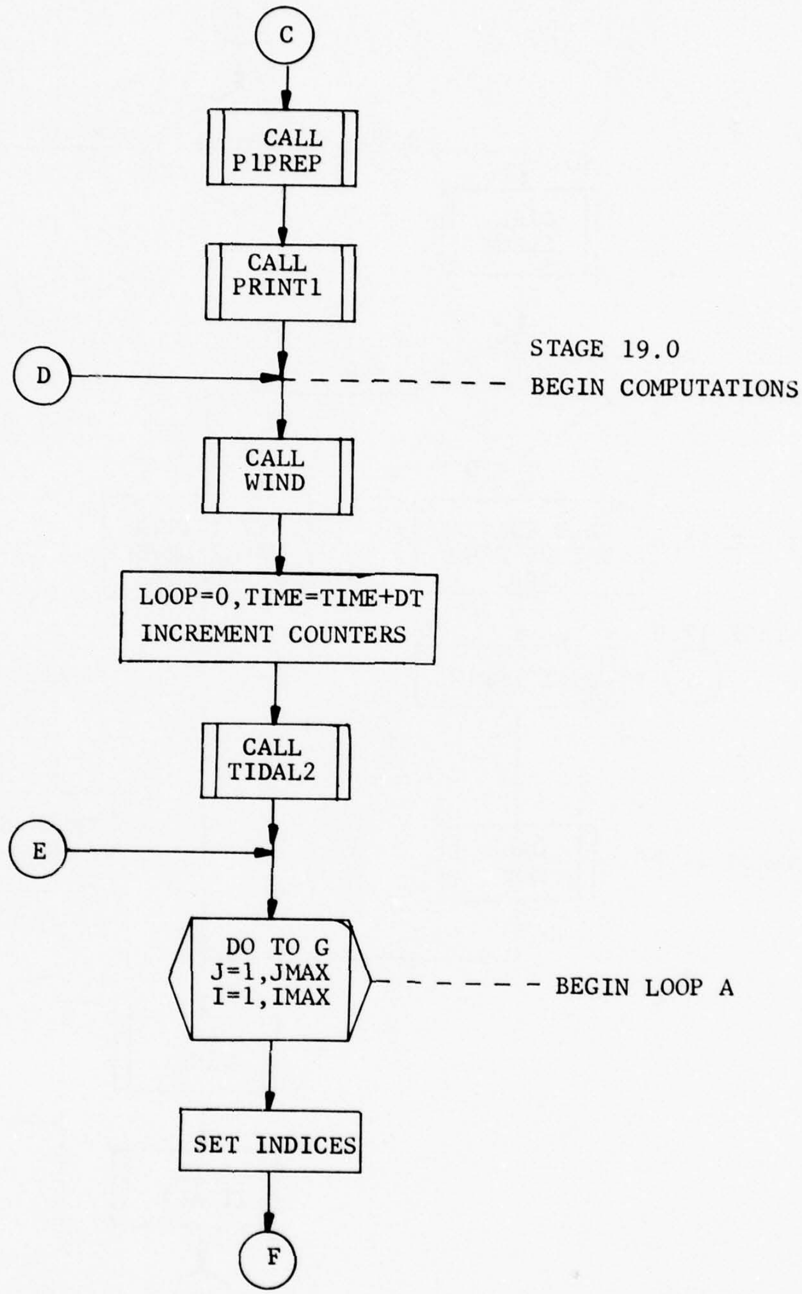


FIG. 5.1 (Continued) FLOW CHART FOR PROGRAM MASON

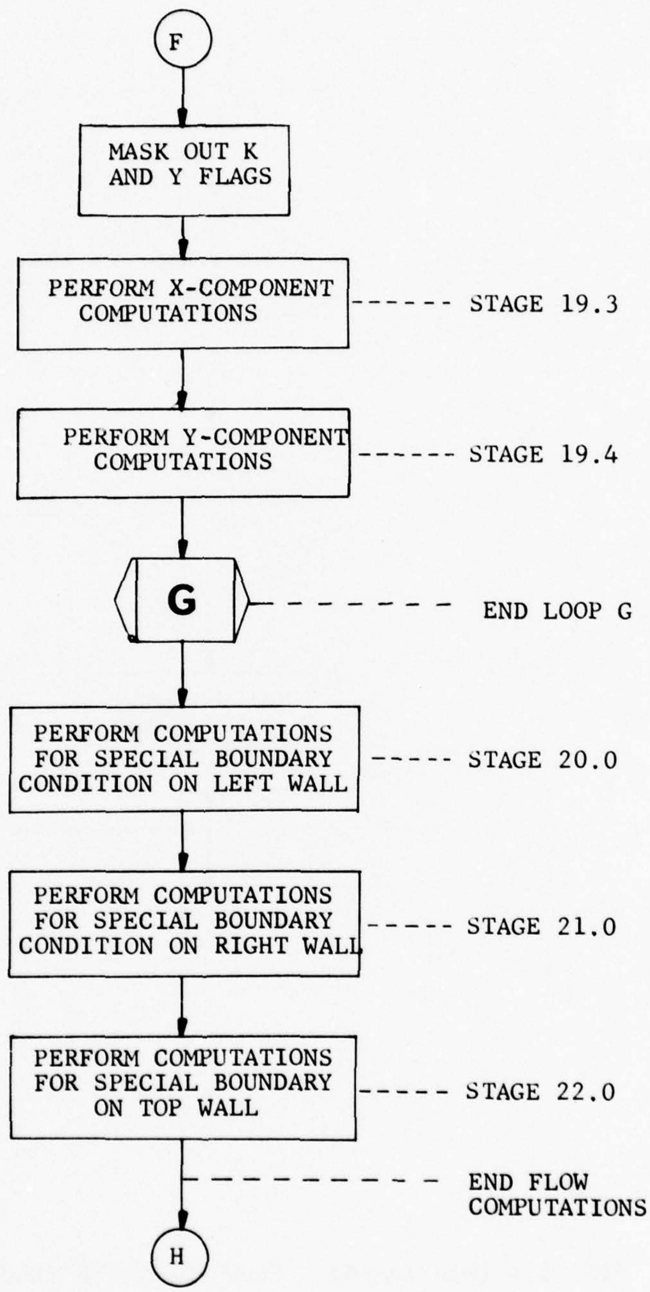


FIG. 5.1 (Continued) FLOW CHART FOR PROGRAM MASON

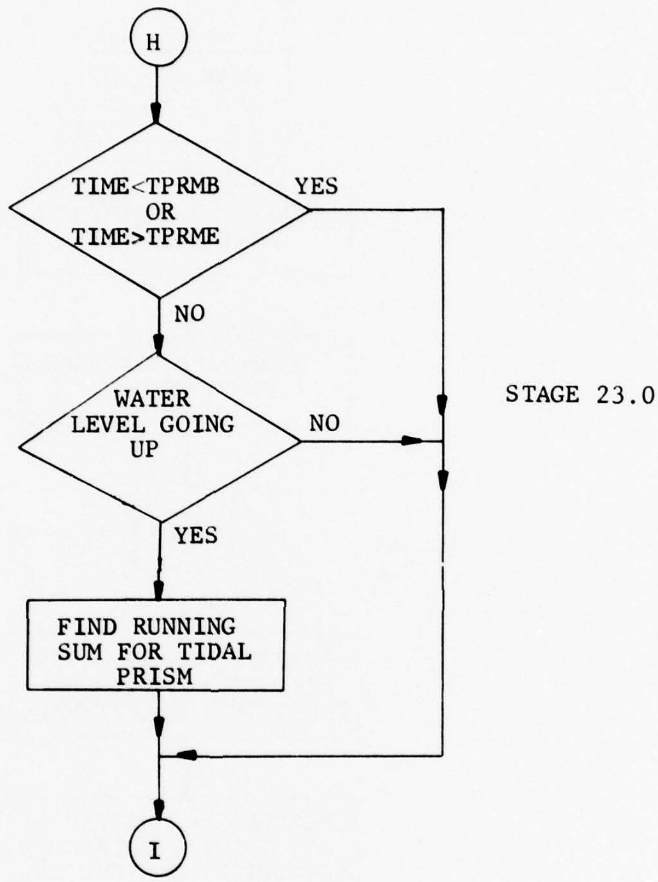


FIG. 5.1 (Continued) FLOW CHART FOR PROGRAM MASON

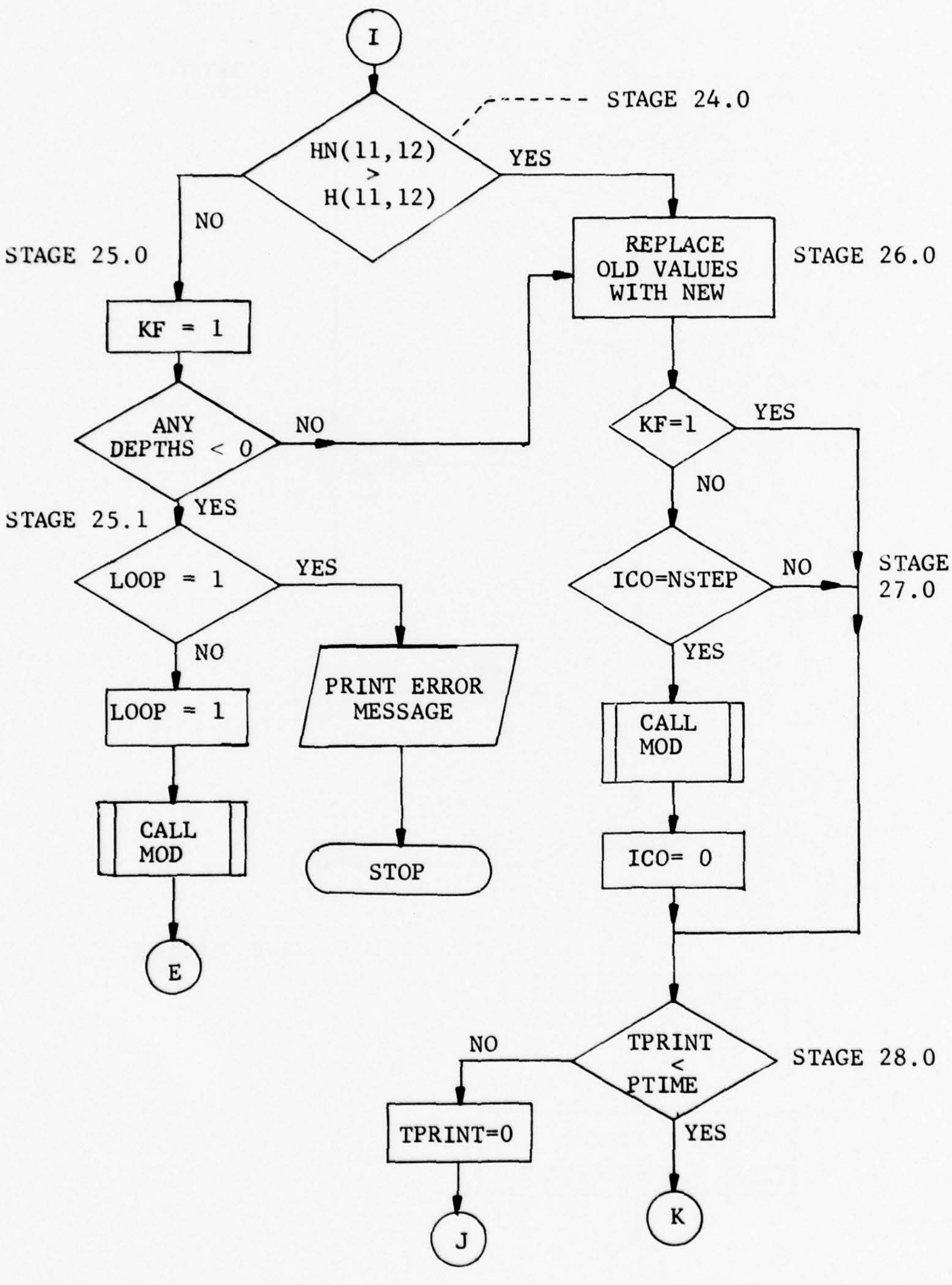


FIG. 5.1 (Continued) FLOW CHART FOR PROGRAM MASON

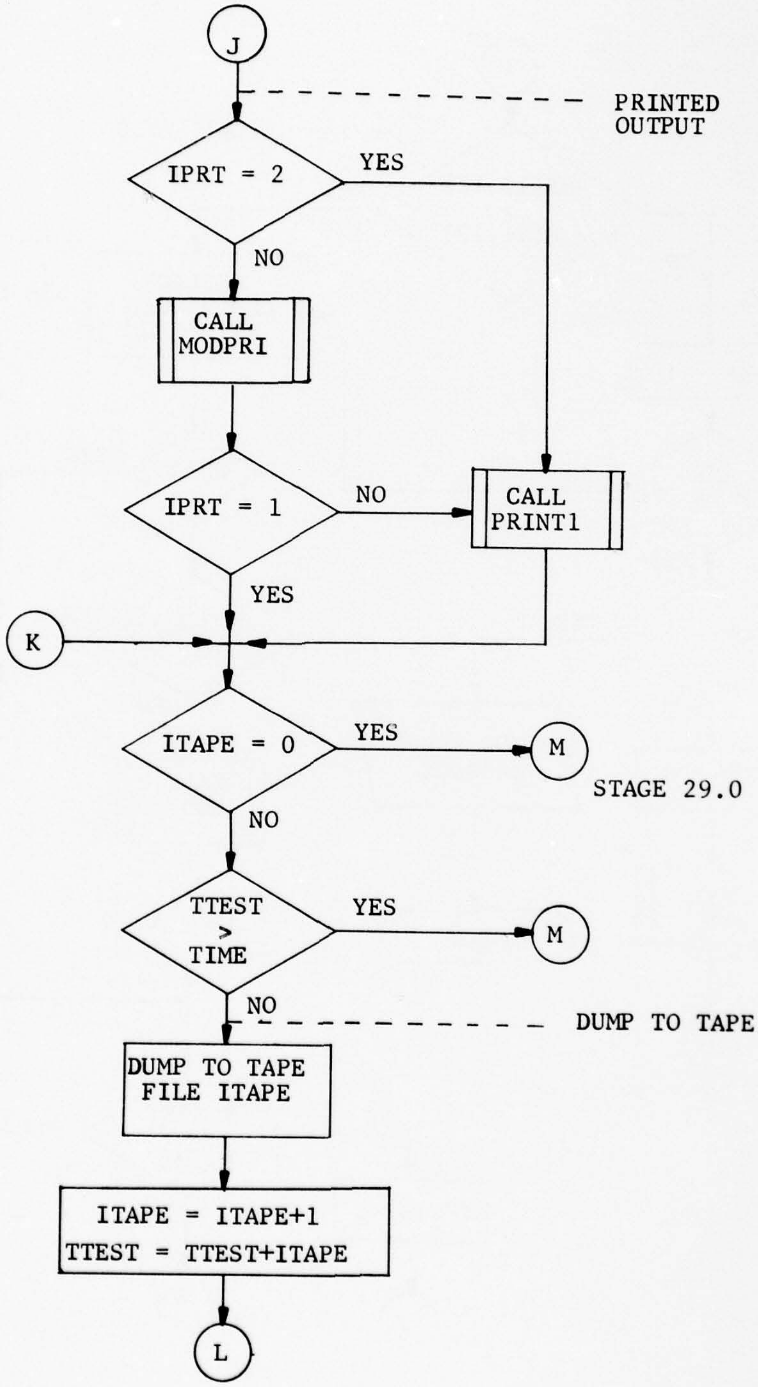


FIG. 5.1 (Continued) FLOW CHART FOR PROGRAM MASON

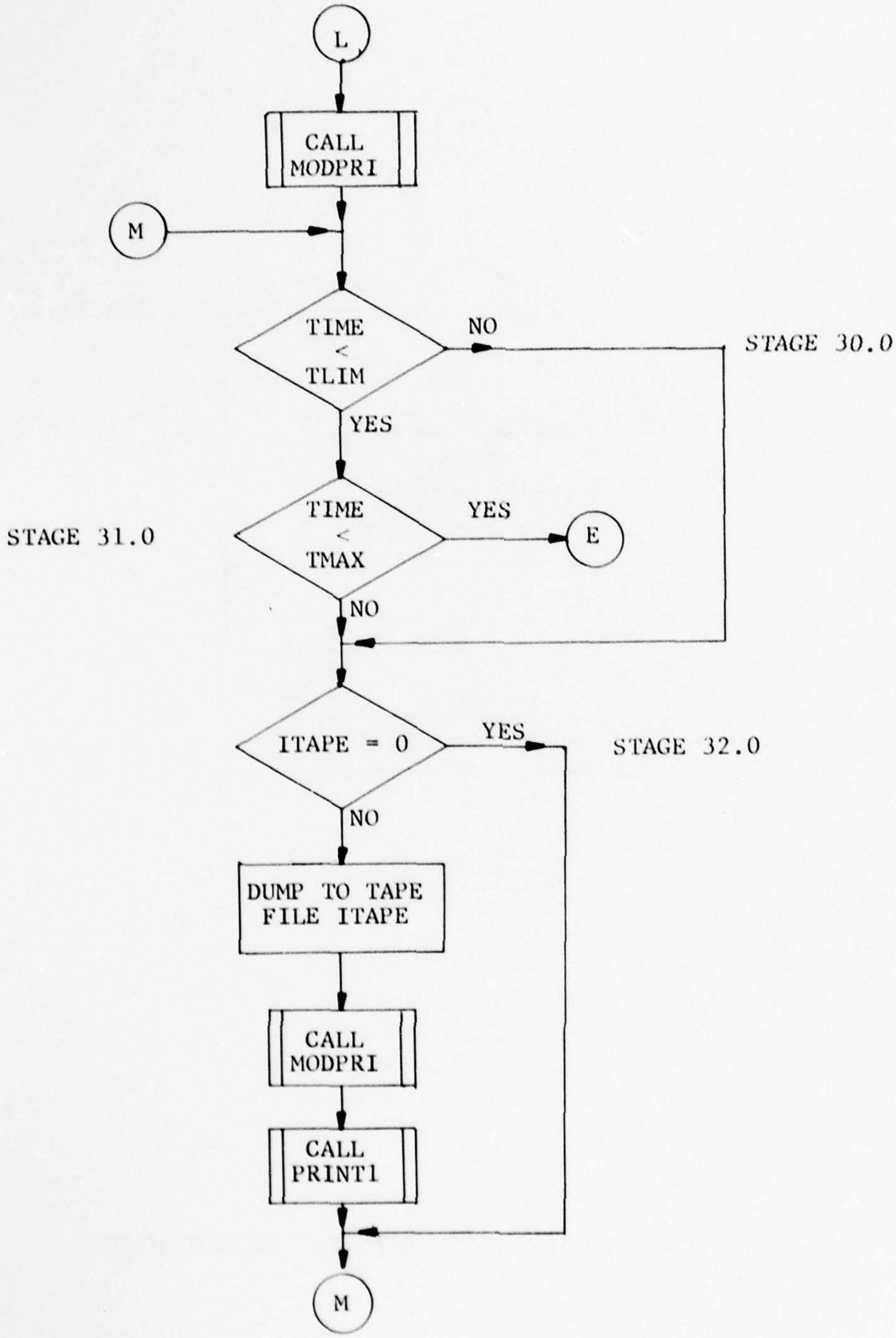


FIG. 5.1 (Continued) FLOW CHART FOR PROGRAM MASON

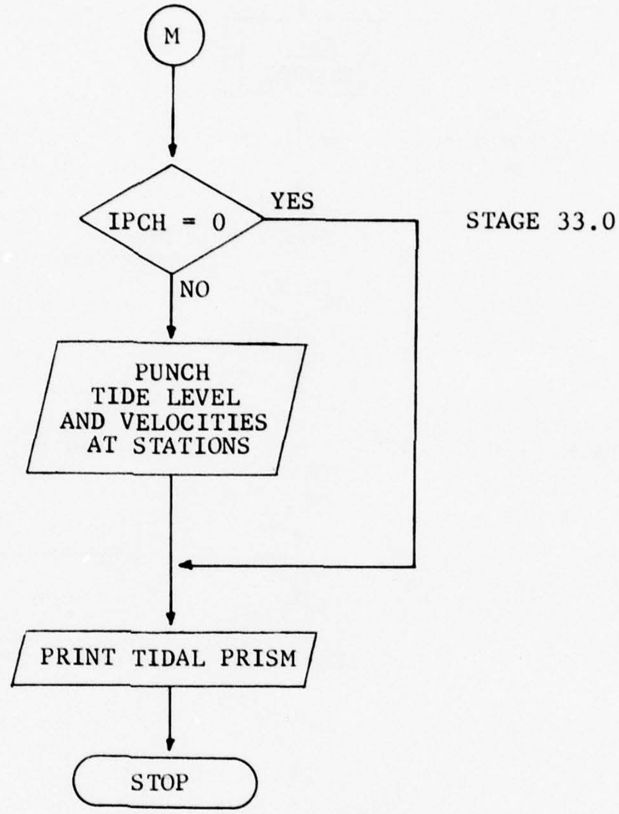


FIG. 5.1 (Continued) FLOW CHART FOR PROGRAM MASON

TABLE 5.2

LISTING OF SUBROUTINE FLAGS

```
      SUBROUTINE FLAGS(LHI,LHJ,LHK)
      DIMENSION LINE(80),LHI(3,25),LHJ(3,25),LHK(3)
      COMMON/A/QX(30,25),QY(30,25),D(30,25),IFLAG(30,25),H(30,25),IMAX,
     1     JMAX,DELTA,JB,JBR,IBU,PUNV(5,4,50),NPU,PCHT(50),IPCH
C
C        ****************************************************************
C        * THIS SUBROUTINE READS THE FLAG CARDS AND SETS UP THE INITIAL *
C        * FLAG FIELD. THE INITIAL FLAG FIELD IS SET UP ASSUMING THAT   *
C        * THE WATER LEVEL (H), IS ZERO (0) FEET W.R.T. TO THE DATUM    *
C        * LEVEL.  ALSO ALL CELLS WITH SPECIAL BOUNDARY CONDITIONS      *
C        * (IFLAG=3) ARE TABULATED AND THEIR SUBSCRIPS STORED AND
C        * PRINTED .                                                    *
C        ****************************************************************
C
C     ******************************************************************
C      STAGE 1 -- INITIALIZE
C     ******************************************************************
C
      DO 500 I=1,3
      DO 500 J=1,25
      LHI(I,J)=0
  500 LHJ(I,J)=0
      IMAXM1=IMAX-1
      JMAXM1=JMAX-1
      IMAX1=IMAX+1
      JMAX1=JMAX+1
      MASK=246
      PRINT 19
   19 FORMAT(1H1,/,'   FLAG X COMPONANT')
C
C     ******************************************************************
C      STAGE 2.0 - READ AND ECHO PRINT X FLAGS
C                  THEN SET UP X COMPONENT FLAGS IN FLAG FIELD
C     ******************************************************************
C
      DO 10 J=1,JMAX
      READ 1,(LINE(I),I=1,IMAX1)
      PRINT 2,(LINE(I),I=1,IMAX1)
    2 FORMAT(5X,80I1)
```

TABLE 5.2 - Continued

LISTING OF SUBROUTINE FLAGS

```
    1 FORMAT(80I1)
      DO 10 I=1,IMAX
      I1=I+1
      IFLAG(I,J)=0
C
C     ******************************************************************
C     STAGE 2.1 - SET FLAG FOR RIGHT HAND WALL
C     ******************************************************************
C
      IFLAG(I,J)=LINE(I1)
C
C     ******************************************************************
C     STAGE 2.2 - IF LEFT HAND WALL CANNOT BE A BOUNDARY OR BARRIER
C                 SET FLAG FOR LEFT WALL = 0 , IF IT CAN SET FLAG
C                 FOR LEFT WALL =1
C     ******************************************************************
C
      IF ( LINE(I) .EQ. 0 .OR. LINE(I) .EQ. 4 ) GO TO 10
      IF ( LINE(I) .EQ. 3 .AND. LINE(I1) .NE. 6 ) GO TO 10
      IFLAG(I,J)=IFLAG(I,J) .OR. 64
   10 CONTINUE
C
C    *******************************************************************
C     STAGE 3.0 - READ AND ECHO PRINT Y FLAGS
C                 THEN SET UP Y COMPONENT FLAGS IN FLAG FIELD
C    *******************************************************************
C
      PRINT 3
    3 FORMAT(1H1,/,'     FLAG Y COMPONANT')
      DO 20 I=1,IMAX
      READ 1,(LINE(J),J=1,JMAX1)
      PRINT 2,(LINE(J),J=1,JMAX1)
      DO 20 J=1,JMAX
      J1=J+1
C
C     ******************************************************************
C     STAGE 3.1 - SET FLAG FOR TOP WALL
C     ******************************************************************
C
      FLD(30,3,IFLAG(I,J))=LINE(J1)
C
C     ******************************************************************
C     STAGE 3.2 - IF BOTTOM WALL CANNOT BE A BOUNDARY OR BARRIER
```

TABLE 5.2 - Continued

LISTING OF SUBROUTINE FLAGS

```
C               SET FLAG FOR BOTTOM EALL = 0, IF IT CAN SET FLAG
C               FOR BOTTOM WALL = 1.
C     ******************************************************************
C
      IF ( LINE(J) .EQ. 0 .OR. LINE(J) .EQ. 4 ) GO TO 20
      IF ( LINE(J) .EQ. 3 .AND. LINE(J1) .NE. 6 ) GO TO 20
      IFLAG(I,J)=IFLAG(I,J) .OR. 128
   20 CONTINUE
C
C    *******************************************************************
C     STAGE 4.0 - FIND AND TABULATE SPECIAL BOUNDARIES
C                 BIAS OF 256 IS ADDED IN HERE ALSO
C    *******************************************************************
C
      K1=0
      K2=0
      K3=0
      LHK(1)=0
      LHK(2)=0
      LHK(3)=0
      DO 30 J=1,JMAX
      DO 30 I=1,IMAX
C
C    *******************************************************************
C     STAGE 4.1.0 - ADD IN BIAS OF 256
C    *******************************************************************
C
      IFLAG(I,J)=IFLAG(I,J) .OR. 256
C
C    *******************************************************************
C     STAGE 4.2.0 - CHECK FOR SPECIAL BOUNDARY ON RIGHT ( LEFT)
C                   HAND WALL
C    *******************************************************************
C
      IF ( IFLAG(I,J) .EQ. 283 ) GO TO 28
      IF ( FLD(33,3,IFLAG(I,J)) .NE. 3 ) GO TO 29
      IF ( FLD(33,3,IFLAG(I-1,J)) .LT. 6  .AND. I .NE. 1 ) GO TO 28
C
C    *******************************************************************
C     STAGE 4.2.1 - STORE INDICES FOR LEFT BOUNDARY
C    *******************************************************************
C
      LHK(1)=LHK(1)+1
```

TABLE 5.2 - Continued

LISTING OF SUBROUTINE FLAGS

```
      K1=K1+1
      LHI(1,K1)=I
      LHJ(1,K1)=J
C
C     *******************************************************************
C     STAGE 4.2.2 - CHECK FOR DOUBLE SPECIAL BOUNDARIES
C     *******************************************************************
C
      IF ( FLD(30,3,IFLAG(I+1,J)) .EQ. 3 ) GO TO 290
      GO TO 30
C
C     *******************************************************************
C     STAGE 4.2.3 - STORE INDICES FOR RIGHT BOUNDARY
C     *******************************************************************
C
   28 LHK(2)=LHK(2)+1
      K2=K2+1
      LHI(2,K2)=I
      LHJ(2,K2)=J
C     *******************************************************************
C     STAGE 4.2.4 - CHECK FOR DOUBLE SPECIAL BOUNDARIES
C     *******************************************************************
C
      IF ( IFLAG(I,J) .EQ. 283 ) GO TO 290
      GO TO 30
C
C     *******************************************************************
C     STAGE 4.3.0 - CHECK FOR SPECIAL BOUNDARY ON TOP WALL
C     *******************************************************************
C
   29  IF ( FLD(30,3,IFLAG(I,J)) .NE. 3 ) GO TO 30
  290 K3=K3+1
      LHK(3)=LHK(3)+1
      LHI(3,K3)=I
      LHJ(3,K3)=J
   30 CONTINUE
      K=K1
C
C     *******************************************************************
C     STAGE 4.4.0 - PRINT TABLE OF INDICES OF SPECIAL CELLS
C     *******************************************************************
C
      IF ( K2 .GT. K ) K=K2
```

TABLE 5.2 - continued

LISTING OF SUBROUTINE FLAGS

```
      IF ( K3 .GT. K ) K=K3
      PRINT 100
      PRINT 31,((J,LHI(1,J),LHJ(1,J),LHI(2,J),LHJ(2,J),LHI(3,J),LHJ(3,J)
     *),J=1,K)
  100 FORMAT(1H1//)
   31 FORMAT(//8X,'SPECIAL CELLS (WALLS AFFECTED)',/10X,'RIGHT     LEFT       TOP
     *     TOP',/5X,'N',4X,3('I    J',5X),(/2I6,I4,I6,I4,I6,I4))
C
C     ******************************************************************
C     STAGE 5.0.0 - ASSUMING WATER LEVEL IS ZERO (0) DETERMAIN
C                   WHICH CELLS ARE FLOODED AND WHICH ARE NOT AND SET
C                   FLAGS ACCORDINGLY.  ALSO REMOVE BIAS FROM CELLS NOT
C                   ALWAYS FLOODED OR ALWAYS DRY.
C
C     ******************************************************************
C
C     STAGE 5.1.0 - CHECK ALL CELLS BUT THOES ALONG ROW WITH J=JMAX
C
C     ******************************************************************
C
      DO 40 J=2,JMAXM1
C
C     ******************************************************************
C     STAGE 5.1.1 - CHECK CELLS IN COLUMN WITH I=IMAX
C                   IF D .GT. 0 CELL ALWAYS FLOODED OR ALWAYS DRY
C     ******************************************************************
C
      IF ( D(IMAX,J) .GT. 0. ) GO TO 36
      IF ( FLD(33,3,IFLAG(IMAX,J)) .NE. 3 ) GO TO 36
C
C     ******************************************************************
C     STAGE 5.1.1.1 - CELL IS DRY - SET APPROPAITE FLAGS
C     ******************************************************************
C
      IFLAG(IMAX,J ) = 237
      IFLAG(IMAX-1,J)=IFLAG(IMAX-1,J) .OR. 5
      IFLAG(IMAX,J-1)=IFLAG(IMAX,J-1) .OR. 40
C
C     ******************************************************************
C     STAGE 5.1.2 - CHECK REST OF CELLS
C     ******************************************************************
C
   36 DO 40 I=2,IMAXM1
```

TABLE 5.2 - continued

LISTING OF SUBROUTINE FLAGS

```
C
C     ******************************************************************
C     STAGE 5.1.2.1 - IF D .GT. 0 CELL ALWAYS FLOODED OR ALWAYS DRY
C     ******************************************************************
C
      IF ( D(I,J) .GT. 0. ) GO TO 40
      IF ( IFLAG(I,J) .EQ. 246 ) GO TO 40
C
C     ******************************************************************
C     STAGE 5.1.2.2 - CHECK FOR SPECIAL BOUNDRY ON TOP OR RIGHT WALL
C     ******************************************************************
C
      IF ( FLD(33,3,IFLAG(I,J)) .EQ. 3 ) GO TO 43
      IF ( FLD(30,3,IFLAG(I,J)) .EQ. 3 ) GO TO 44
      IFLAG(I,J)=IFLAG(I,J) .OR. 237
      IFLAG(I,J)=IFLAG(I,J)-256
C
C     ******************************************************************
C     STAGE 5.1.2.3 - SET FLAG IN CELL AND ADJACENT CELLS TO  INDICATE
C                     DRY LAND AND REMOVE BIAS IN CELLS WITHOUT SPECIAL
C                     BOUNDRIES
C     ******************************************************************
C
      IF ( IFLAG(I-1,J) .EQ. 499 ) GO TO 41
      IFLAG(I-1,J)=IFLAG(I-1,J) .OR. 5
   41 IFLAG(I+1,J)=IFLAG(I+1,J) .OR. 64
      IFLAG(I,J-1)=IFLAG(I,J-1) .OR. 40
   42 IFLAG(I,J+1)=IFLAG(I ,J+1) .OR. 128
      GO TO 40
C
C     ******************************************************************
C     STAGE 5.1.2.4 - SET FLAGS TO INDICATE DRY LAND IN CELL WTIH
C                     SPECIAL BOUNDRIES AND REMOVE BIAS
C     ******************************************************************
C
   43 IFLAG(I,J)=IFLAG(I,J) .AND. 504
      IF ( FLD(30,3,IFLAG(I,J)) .NE. 3 ) GO TO 45
   44 IFLAG(I,J)=IFLAG(I,J) .AND. 455
   45 IFLAG(I,J)=IFLAG(I,J) .OR. 237
      IFLAG(I-1,J)=IFLAG(I-1,J) .OR. 5
      IFLAG(I,J-1)=IFLAG(I,J-1) .OR. 40
      IFLAG(I,J)=IFLAG(I,J)-256
   40 CONTINUE
```

TABLE 5.2 - continued

LISTING OF SUBROUTINE FLAGS

```
C
C     ******************************************************************
C     STAGE 5.2.0 - CHECK CELLS ALONG ROW WITH J=JMAX
C     ******************************************************************
C
      DO 50 I=2,IMAXM1
C
C     ******************************************************************
C     STAGE 5.2.1 - IF D .GT. 0 CELL ALWAYS FLOODED OR ALWAYS DRY
C     ******************************************************************
C
      IF ( D(I,JMAX) .GT. 0. ) GO TO 50
      IF ( FLD(30,3,IFLAG(I,JMAX)) .NE. 3 ) GO TO 50
      IFLAG(I,JMAX)=237
      IFLAG(I,JMAX-1)=IFLAG(I,JMAX-1) .OR. 40
      IFLAG(I-1,JMAX)=IFLAG(I-1,JMAX) .OR. 5
      IFLAG(I+1,JMAX)=IFLAG(I+1,JMAX) .OR. 64
   50 CONTINUE
C
C     ******************************************************************
C     STAGE 5.3.0 - CHECK CELL IMAX,JMAX
C     ******************************************************************
C
      IF ( D(IMAX,JMAX) .GT. 0.)GO TO 60
      IFLAG(IMAX,JMAX)=237
      IFLAG(IMAXM1,JMAX)=IFLAG(IMAXM1,JMAX) .OR. 5
      IFLAG(IMAX,JMAXM1)=IFLAG(IMAX,JMAXM1) .OR. 40
C
C     ******************************************************************
C     STAGE 6.0.0 - SET DEPTHS AND WATER LEVELS IN  PERMANENTLY
C                   DRY CELLS TO -1000.
C     ******************************************************************
C
   60 DO 70 J=1,JMAX
      DO 70 I=1,IMAX
      IF ( IFLAG(I,J) .NE. 253 ) GO TO 64
      D(I,J)=-1000.
      H(I,J)=-1000.
      GO TO 70
   64 IF ( IFLAG(I,J) .NE. 502 .AND. IFLAG(I,J) .NE. 503 ) GO TO 65
      IF ( FLD(33,3,IFLAG(I-1,J)) .EQ. 2 ) GO TO 70
      D(I,J)=-1000.
      H(I,J)=-1000.
```

TABLE 5.2 - continued

LISTING OF SUBROUTINE FLAGS

```
      GO TO 70
   65 IF ( IFLAG(I,J) .NE. 510 .AND. IFLAG(I,J) .NE. 511 ) GO TO 70
      D(I,J)=-1000.
      H(I,J)=-1000.
   70 CONTINUE
      RETURN
      END
```

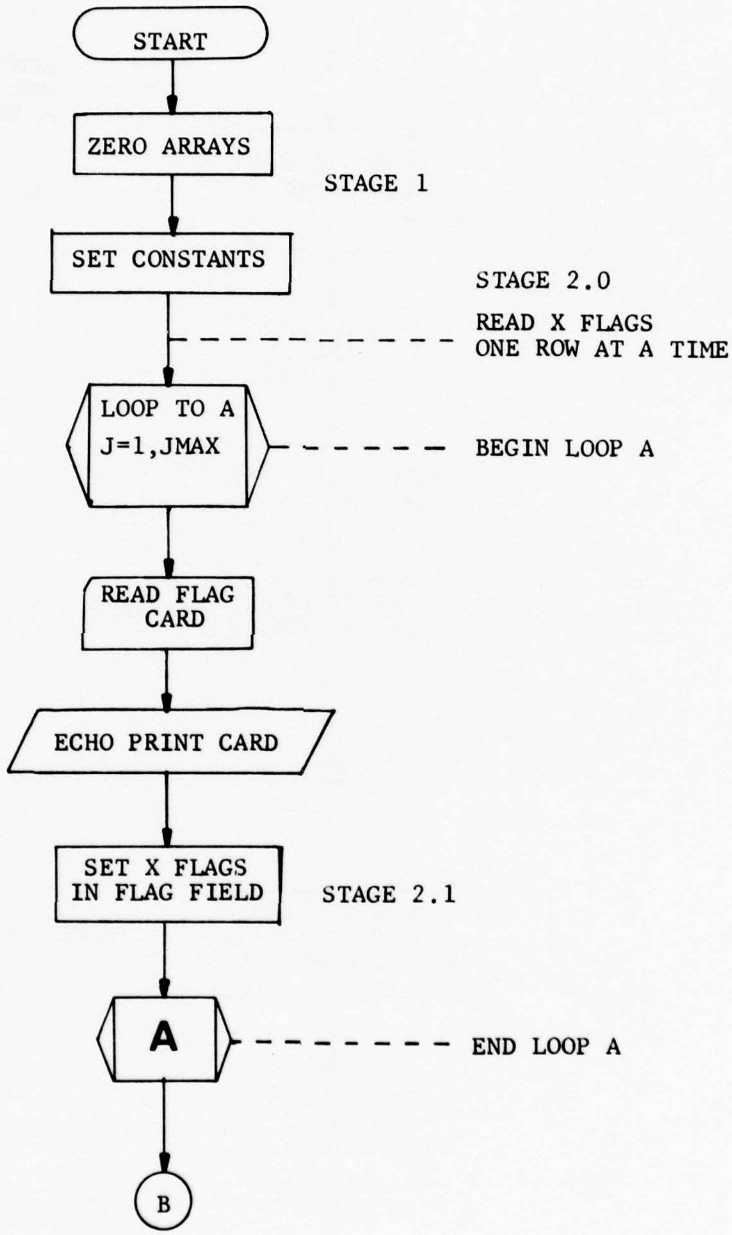


FIG. 5.2 FLOW CHART FOR SUBROUTINE FLAGS

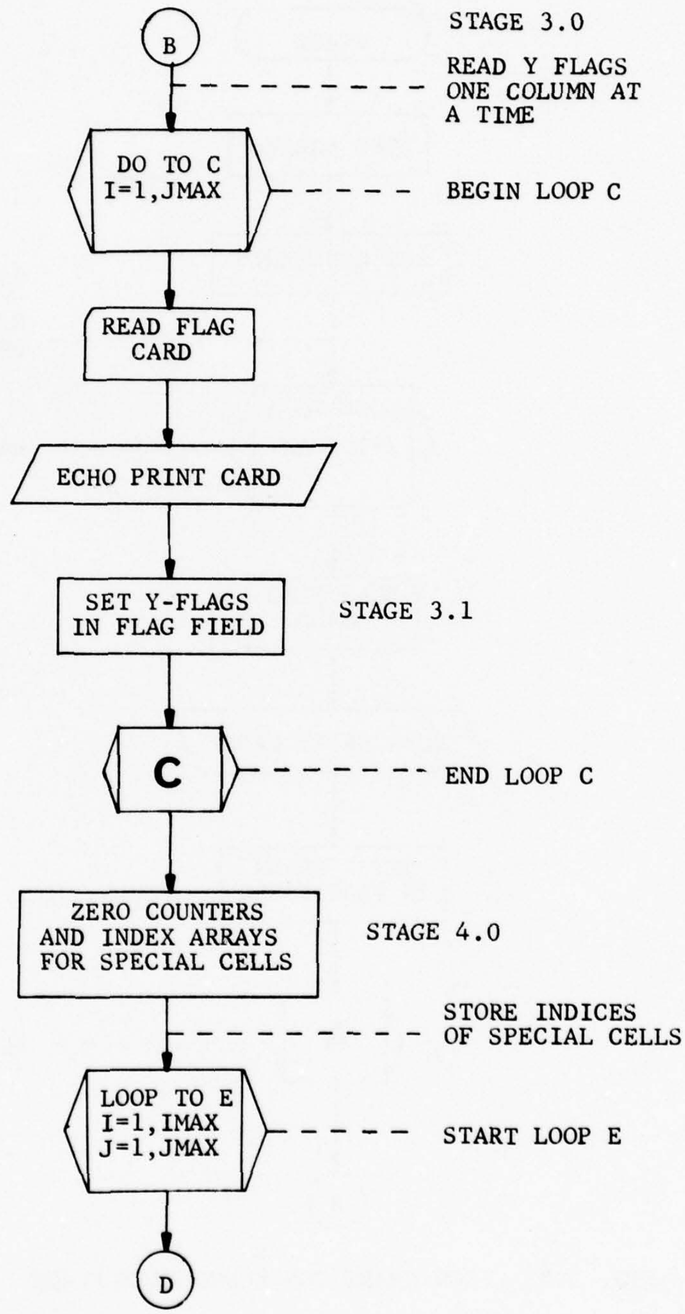


FIG. 5.2(Continued) FLOW CHART FOR SUBROUTINE FLAGS

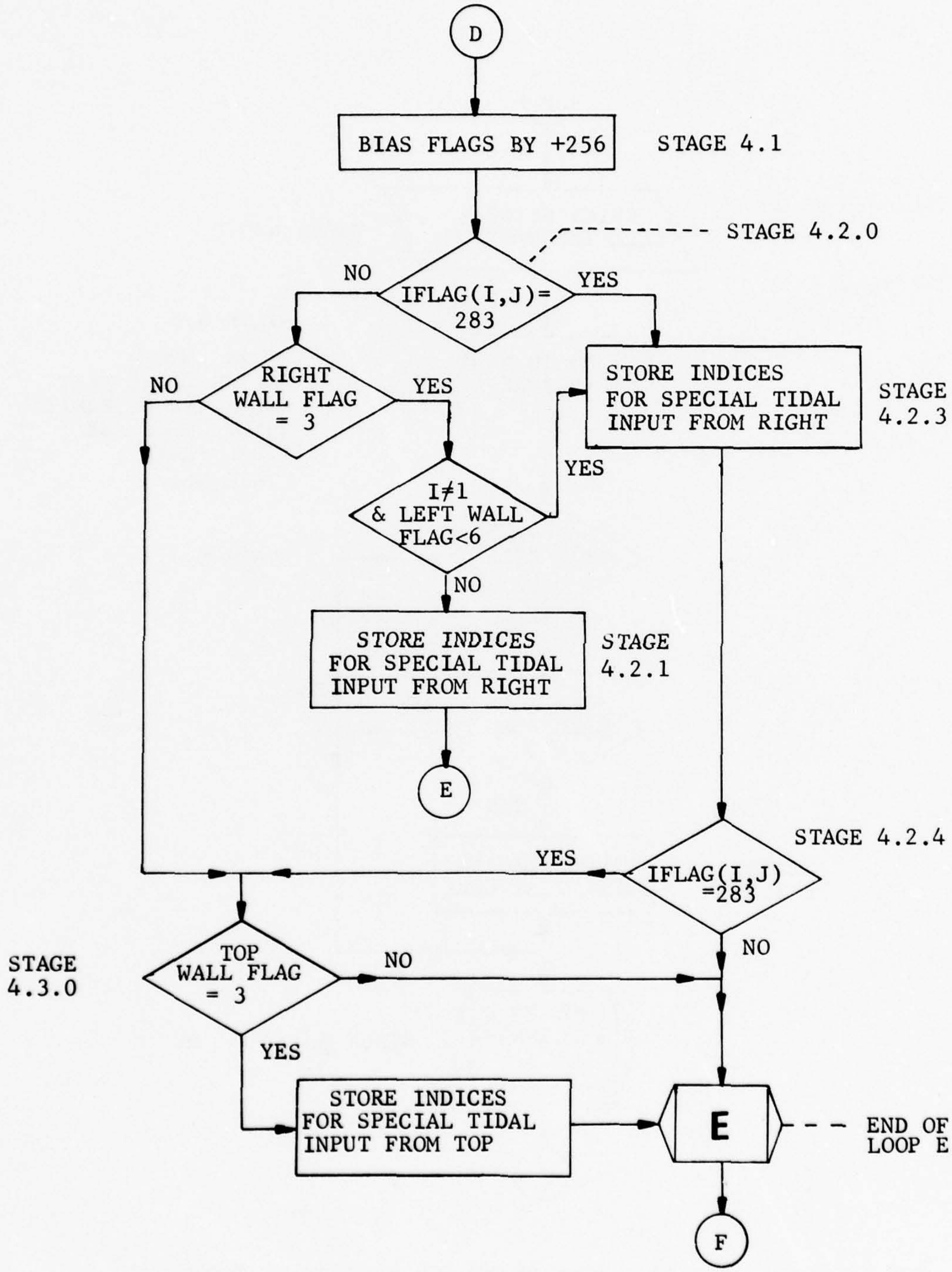


FIG. 5.2 (Continued) FLOW CHART FOR SUBROUTINE FLAGS

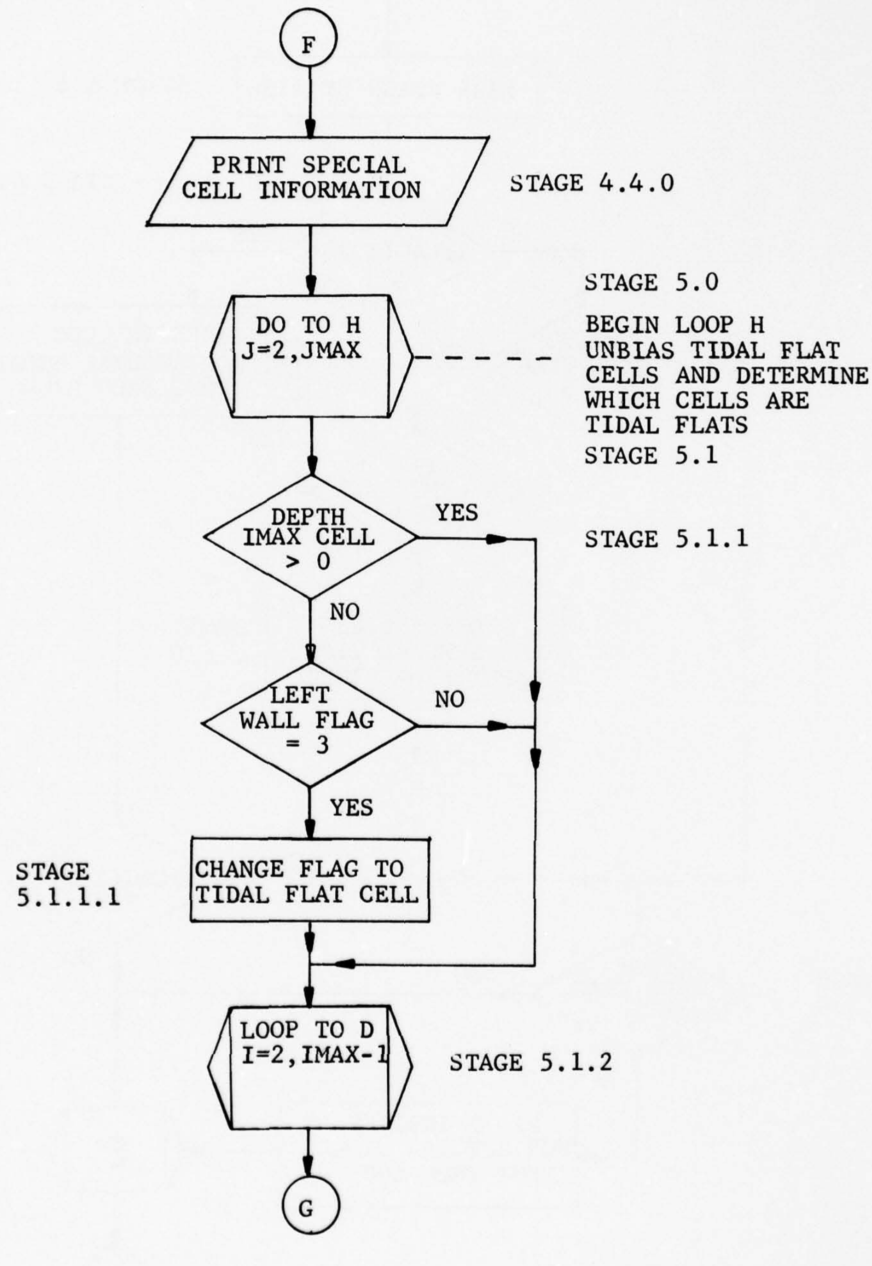


FIG. 5.2 (Continued) FLOW CHART FOR SUBROUTINE FLAGS

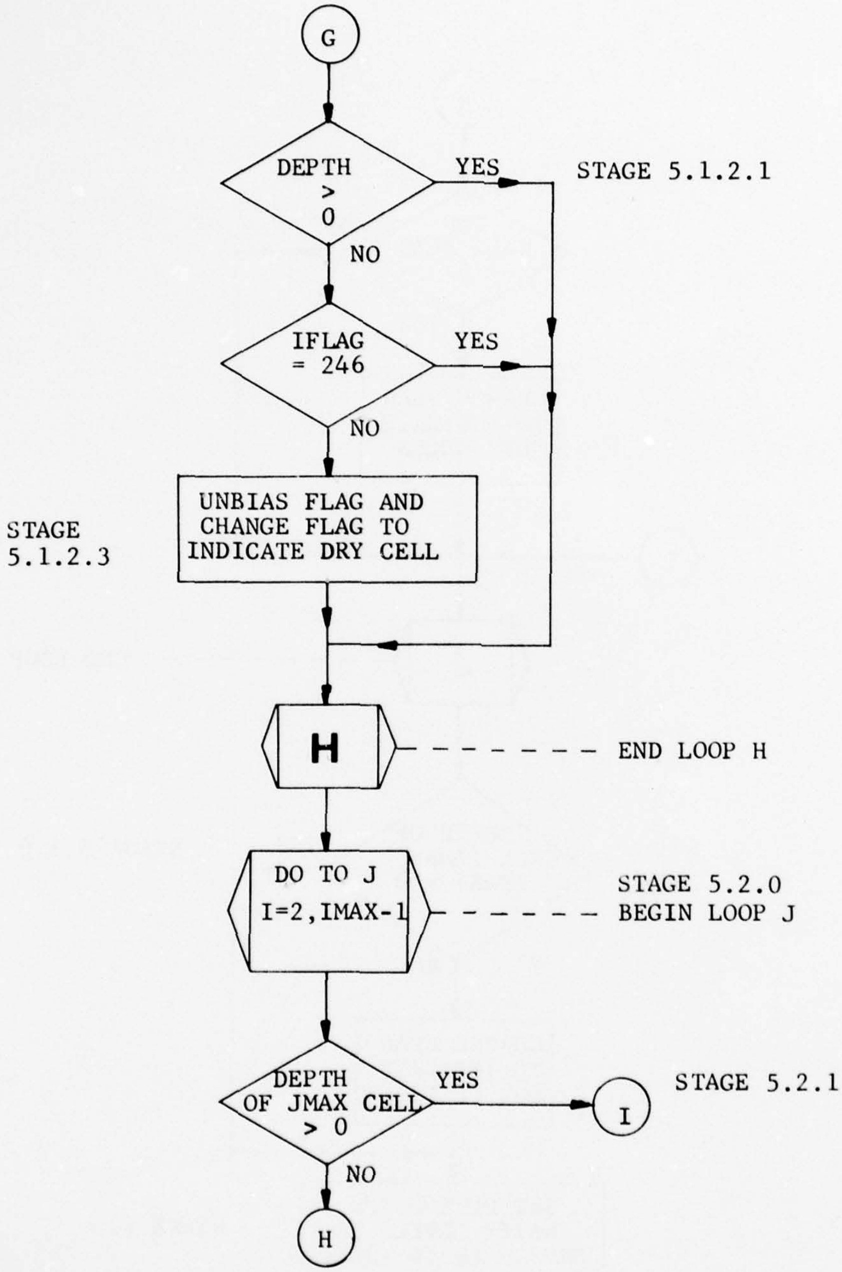


FIG. 5.2 (Continued)   FLOW CHART FOR SUBROUTINE FLAGS

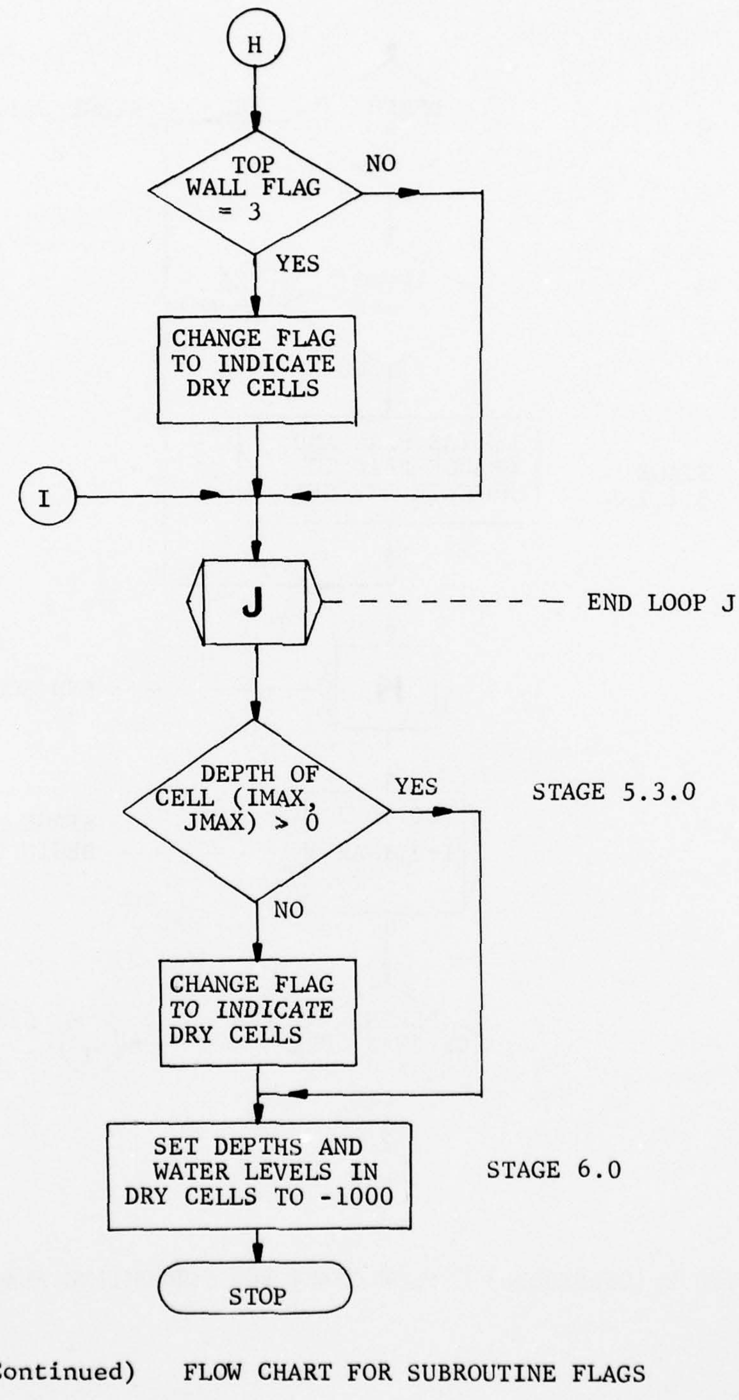


FIG. 5.2 (Continued) FLOW CHART FOR SUBROUTINE FLAGS

TABLE 5.3

LISTING OF SUBROUTINE MOD

```
      SUBROUTINE MOD(TIME,IPRMOD,Z,IOP)
      COMMON/A/QX(30,25),QY(30,25),D(30,25),IFLAG(30,25),H(30,25),IMAX,
     1     JMAX,DELTA,JB,JBR,IBU,PUNV(5,4,50),NPU,PCHT(50),IPCH
      DIMENSION Z(30,25)
      DATA /ITR/-1/
C
C         ************************************************************
C         * THIS SUBROUTINE CHECKS FOR THE FLOODING AND DRYING UP OF    *
C         * CELLS IN THE TIDAL FLATS.                                   *
C         ************************************************************
C
      MASK=237
  799 ITR=ITR+1
      IK=IMAX
      JK=JMAX
      IK1=IK-1
      JK1=JK-1
  803 DELTAS=DELTA
C
C     ****************************************************************
C     STAGE 1.0 - THE LOOP TO 600 CHECKS ALL CELLS BUT THOSE IN THE ROW
C                 WITH J = JMAX AND THE COLUMN WITH I=IMAX
C    *****************************************************************
C
      DO 600 J=JB,JK1
      DO 600 I=2,IK1
      IM1=I-1
      IP1=I+1
      JM1=J-1
      JP1=J+1
C
C     ****************************************************************
C     STAGE 1.1 - CHECK IF CELL IS ALWAYS FLOODED OR ALWAYS DRY
C     ****************************************************************
C
      IF( FLD(27,1,IFLAG(I,J)) .EQ. 1 ) GO TO 600
C
C     ****************************************************************
C     STAGE 1.2 - CHECK IF CELL IS DRY - IF SO JUMP TO CHECK IF
```

TABLE 5.3 - continued

LISTING OF SUBROUTINE MOD

```
C             IF IT FLOODS
C      ****************************************************************
C
       IF ( IFLAG(I,J) .GE. 237 .AND. IFLAG(I,J) .NE. 248 ) GO TO 580
C
C      ****************************************************************
C
C      STAGE 1.3 - SECTION FROM HERE TO COMMENT 'END A' CHECKS
C                  IF CELL DRIES UP
C
C      ****************************************************************
C
C      STAGE 1.3.1 - IF DEPTH IS GREATER THAN DELTA CONSIDER CELL FLOODED
C                    IF DEPTH LESS THAN OR EQUAL TO DELTA CONSIDER CELL
C                    DRIED UP AND SET FLAGS ACCORDINGLY
C
C      ****************************************************************
C
       IF ( D(I,J) .GT. DELTA ) GO TO 600
  568  CONTINUE
   10  FORMAT(5X,'*******',2I3,F6.1,5(3X,O3))
       IF ( FLD(33,3,IFLAG(I,J)).EQ.3) IFLAG(I,J)=IFLAG(I,J).AND.504
       IF ( FLD(30,3,IFLAG(I,J)).EQ. 3)IFLAG(I,J)=IFLAG(I,J).AND.455
       IFLAG(I,J)=IFLAG(I,J) .OR. 237
       IF ( FLD(33,3,IFLAG(IM1,J)) .EQ. 6 ) GO TO 570
       IF ( FLD(33,3,IFLAG(I,J)) .EQ. 3 ) GO TO 569
       IF ( I .EQ. 2 ) GO TO 570
       IFLAG(IM1,J)=IFLAG(IM1,J) .OR. 5
       GO TO 570
  569  H(IM1,J)=-1000.
       D(IM1,J)=-1000.
  570  IF ( FLD(30,3,IFLAG(I,JM1)) .EQ. 6 ) GO TO 575
       IFLAG(I,JM1)=IFLAG(I,JM1) .OR. 40
  575  IFLAG(IP1,J)=IFLAG(IP1,J) .OR. 64
       IFLAG( I,JP1)=IFLAG(I,JP1) .OR. 128
C
C      ****************************************************************
C      STAGE 1.3.2 - SET APPROPRIATE FLOWS TO ZERO (0) AND WATER LEVEL
C                    AND DEPTH TO -1000.
C      ****************************************************************
C
       QX(I,J)=0.
       QY(I,J)=0.
```

TABLE 5.3 - continued

LISTING OF SUBROUTINE MOD

```
      QX(IM1,J)=0.
      QY(I,JM1)=0.
      D(I,J)=-1000.
      H(I,J)=-1000.
   11 FORMAT(21X,5(3X,O3))
C
C     ******************************************************************
C     END A  ---  END OF STAGE 1.3  ---   END A
C     ******************************************************************
C
      GO TO 600
C
C     ******************************************************************
C     STAGE 1.4 - SECTION OF LOOP FROM HERE TO COMMENT 'END G' CHECKS
C                 IF CELL FLOODS AND IF SO SETS DEPTH AND WATER LEVEL
C     ******************************************************************
C
  580 IF ( D(I,J) .LT. -10. ) GO TO 582
      D(I,J)=-1000.
      H(I,J)=-1000.
      GO TO 600
  582 ZZ=Z(I,J)
      IJM=0
      IJM=0
C
C     ******************************************************************
C     STAGE 1.4.1 - CHECK IF CELL FLOODS FORM ANY DIRECTION
C     ******************************************************************
C
      IF ( H(IM1,J)-ZZ .GE. DELTA ) GO TO 581
      IF ( H(I,JM1)-ZZ .GE. DELTA ) GO TO 581
      IF ( H(I,JP1)-ZZ .GE. DELTA ) GO TO 581
      IF ( H(IP1,J)-ZZ .GE. DELTA ) GO TO 581
C
C     ******************************************************************
C     STAGE 1.4.2 - CELL DOES NOT FLOOD - JUMP TO END OF LOOP
C     ******************************************************************
C
      GO TO 600
  581 DELTA = 0.0
C
C     ******************************************************************
C     STAGE 1.4.3 - CODE FROM HERE TO ' END B' CHECKS IF CELL FLOODS
```

TABLE 5.3 - continued

LISTING OF SUBROUTINE MOD

```
C                FROM LEFT AND CA
C                FROM LEFT AND CHANGES APPROPRIATE FLAGS
C     ******************************************************************
C
      IF( FLD(33,3,IFLAG(IM1,J)) .GE. 6 ) GO TO 585
      IF ( (H(IM1,J)-ZZ) .LT. DELTA ) GO TO 585
      IF ( FLD(33,3,IFLAG(IM1,J )) .EQ. 3 ) GO TO 587
      IFLAG(IM1,J)=IFLAG(IM1,J) .AND. 504
  587 CONTINUE
      IFLAG(I,J)=IFLAG(I,J) .AND. 191
      IJM=IJM+1
       IF ( IFLAG(I,J+1) .NE. 502 ) GO TO 583
      IF ( FLD(30,3,IFLAG(I,J)) .EQ. 7 ) GO TO 583
      FLD(30,3,IFLAG(I,J))=3
      GO TO 585
  583 IF ( IFLAG(I+1,J) .NE. 502 ) GO TO 585
      IF ( FLD(33,3,IFLAG(I,J)) .EQ. 7 ) GO TO 585
      FLD(33,3,IFLAG(I,J))=3
C
C     ******************************************************************
C
C     END B  ---  END OF STAGE 1.4.3  ---  END B
C
C     ******************************************************************
C     ******************************************************************
C
C     STAGE 1.4.4 - CODE FROM HERE TO 'END C' CHECKS IF CELL FLOODS
C                   FROM BOTTOM AND CHANGES APPROPRIATE FLAGS
C
C     ******************************************************************
C
  585 IF ( FLD(30,3,IFLAG(I,JM1)) .GE. 6 ) GO TO 590
      IF ( (H(I,JM1)-ZZ) .LT. DELTA ) GO TO 590
      IJM=IJM+1
      IFLAG(I,JM1)=IFLAG(I,JM1) .AND. 455
      IFLAG(I,J)=IFLAG(I,J) .AND. 127
      IJM=IJM+1
      IF ( IFLAG(I,J+1) .NE. 502 ) GO TO 586
      IF ( FLD(30,3,IFLAG(I,J)) .EQ. 7 ) GO TO 586
      FLD(30,3,IFLAG(I,J))=3
      GO TO 590
  586 IF ( IFLAG(I+1,J) .NE. 502 ) GO TO 590
      IF ( FLD(33,3,IFLAG(I,J)) .EQ. 7 ) GO TO 590
```

TABLE 5.3 - continued

LISTING OF SUBROUTINE MOD

```
      FLD(33,3,IFLAG(I,J))=3
C
C     ******************************************************************
C
C     END C  ---  END OF STAGE 1.4.4 ---  END C
C
C     ******************************************************************
C     ******************************************************************
C
C     STAGE 1.4.5 - CODE FROM HERE TO 'EDN D' CHECKS IF CELL FLOODS
C                   FROM RIGHT AND CHANGES APPROPRIATE FLAGS
C
C     ******************************************************************
C
  590 IF ( FLD(33,3,IFLAG(I,J)) .GE. 6 ) GO TO 595
      IF ( (H(IP1,J)-ZZ) .LT. DELTA ) GO TO 595
      IFLAG(IP1,J)=IFLAG(IP1,J) .AND. 447
      IFLAG(I,J)=IFLAG(I,J) .AND. 248
      IJM=IJM+1
      IF ( IFLAG(I,J+1) .NE. 502 ) GO TO 595
      IF ( FLD(30,3,IFLAG(I,J)) .EQ. 7 ) GO TO 595
      FLD(30,3,IFLAG(I,J))=3
C
C     ******************************************************************
C
C     END D  ---  END OF STAGE 1.4.5  ---  END D
C
C     ******************************************************************
C     ******************************************************************
C
C     STAGE 1.4.6 - CODE FROM HERE TO 'END E' CHECKS IF CELL FLOODS
C                   FROM TOP AND CHANGE APPROPRIATE FLAGS
C
C     ******************************************************************
C
  595 IF( FLD(30,3,IFLAG(I,J)) .GE. 6 ) GO TO 599
      IF ( ( H(I,JP1)-ZZ) .LT. DELTA ) GO TO 599
      IFLAG(I,JP1)=IFLAG(I,JP1) .AND. 383
      IFLAG(I,J)=IFLAG(I,J) .AND. 199
      IJM=IJM+1
      IF ( IFLAG(I+1,J) .NE. 502 ) GO TO 599
      IF ( FLD(33,3,IFLAG(I,J)) .EQ. 7 ) GO TO 599
      FLD(33,3,IFLAG(I,J))=3
```

TABLE 5.3 - continued

LISTING OF SUBROUTINE MOD

```
C
C     ****************************************************************
C
C     END E  ---  END OF STAGE 1.4.6  ---  END E
C
C     ****************************************************************
C     ****************************************************************
C
C     STAGE 1.4.7 - CODE FROM HERE TO 'END F' FINDS WATER LEVEL IN
C                   FLOODED CELL. WATER LEVEL SET TO AVERAGE
C                   LEVEL OF ADJACENT FLOODED CELLS.
C
C     ****************************************************************
C
  599 IF( ABS(DELTA) .GT. .0001 ) GO TO 600
      IC=0
      SUM=0.
      IF ( D(IM1,J) .LT. 0. ) GO TO 400
      IC=1
      SUM=H(IM1,J)
  400 IF ( D(I,JM1) .LT. 0. ) GO TO 401
      IC=IC+1
      SUM=SUM+H(I,JM1)
  401 IF ( D(IP1,J) .LT. 0. ) GO TO 402
      IC=IC +1
      SUM=SUM+H(IP1,J)
  402 IF ( D(I,JP1) .LT. 0. ) GO TO 403
      IC=IC+1
      SUM=SUM+H(I,JP1)
  403 SUM=SUM/IC
      H(I,J)=SUM
      D(I,J)=SUM-ZZ
      DELTA=DELTAS
      IF ( IC .GT. IJM ) GO TO 568
      IF ( D(I,J) .LE. 0. ) GO TO 568
C     ****************************************************************
C     STAGE 1.4.8 - CHECK FOR TRACE BACKC
C     ****************************************************************
      IF ( SUM .GT. Z(I,JM1) .AND. D(I,JM1) .LT. 0.) GO TO 405
      IF ( SUM .GT. Z(IM1,J) .AND. D(IM1,J) .LT. 0. ) GO TO 405
      GO TO 600
  405 CALL TRACE(I,J,Z)
C     ****************************************************************
```

TABLE 5.3 - continued

LISTING OF SUBROUTINE MOD

```
C
C     END G    END G    END G    END G    END G
C
C     ******************************************************************
C
  600 CONTINUE
C
C     ******************************************************************
C
C     STAGE 2.0 - THE LOOP TO 640 CHECKS CELLS IN THE COLUMN WITH I=IMAX
C
C     ******************************************************************
C
      DELTA = DELTAS
      IF( JBR .EQ. 0 ) GO TO 455
      DO 640 J=JBR,JK1
      JM1=J-1
      JP1=J+1
C     ******************************************************************
C     STAGE 2.1 - CHECK IF CELL IS ALWAYS FLOODED OR ALWAYS DRY
C     ******************************************************************
C
      IF ( FLD(27,1,IFLAG(IK,J)) .EQ. 1 ) GO TO 640
C
C     ******************************************************************
C     STAGE 2.2 - IS CELL DRY - IF SO JUMP TO SEE IF IT FLOODS
C     ******************************************************************
C
      IF ( IFLAG(IK,J) .GE. 237 ) GO TO 605
C
C     ******************************************************************
C     STAGE 2.3 - THE CODE FROM HERE TO 'END J' CHECKS IF THE CELL
C                 DRIES UP AND CHANGES THE FLAGS ACCORDINGLY
C     ******************************************************************
C
      IF ( D(IK,J) .GT. DELTA ) GO TO 640
  601 IFLAG(IK,JP1)=IFLAG(IK,JP1) .OR. 128
      IFLAG(IK,JM1)=IFLAG(IK,JM1) .OR. 40
      IFLAG(IK1,J)=IFLAG(IK1,J) .OR. 5
      IF (FLD(33,3,IFLAG(IK,J)).EQ. 3 ) IFLAG(IK,J)=IFLAG(IK,J).AND. 504
      IF ( FLD(30,3,IFLAG(IK,J)) .EQ. 3 )IFLAG(IK,J)=IFLAG(IK,J).AND.455
      IFLAG(IK,J)=IFLAG(IK,J) .OR. MASK
C
```

TABLE 5.3 - continued

LISTING OF SUBROUTINE MOD

```
C     *****************************************************************
C     STAGE 2.3.1 - SET APPROPRIATE FLOWS TO ZERO (0) AND DEPTHS
C                   AND WATER LEVELS TO -1000.
C     *****************************************************************
C
  602 H(IK,J)=-1000.
      D(IK,J)=-1000.
      QX(IK1,J)=0.
      QX(IK,J)=0.
      QY(IK,J)=0.
      QY(IK,JM1)=0.
      GO TO 640
C
C     *****************************************************************
C
C     END J  ---  END OF STAGE 2.3  ---  END J
C
C     *****************************************************************
C     *****************************************************************
C
C      STAGE 2.4.0 - THE CODE FROM HERE TO END 0 CHECKS IF CELL
C                    FLOODS AND CHANGES FLAGS AND DEPTHS ACCORDINGLY
C
C     *****************************************************************
C
  605 IJM=0
      ZZ=Z(IK,J)
C
C     *****************************************************************
C     STAGE 2.4.1 - CHECK IF CELL FLOODS FROM ANY DIRECTION
C     *****************************************************************
C
      IF ((H(IK1,J) - ZZ) .GT. DELTA ) GO TO 604
      IF ( (H(IK,JM1)-ZZ) .GT. DELTA ) GO TO 604
      IF ((H(IK,JP1)-ZZ) .GT. DELTA ) GO TO 604
C
C     *****************************************************************
C     STAGE 2.4.2 - CELL DOES NOT FLOOD - JUMP TO END OF LOOP
C     *****************************************************************
C
      GO TO 640
  604 DELTA = 0.0
C
```

TABLE 5.3 - continued

LISTING OF SUBROUTINE MOD

```
C     ********************************************************************
C
C     STAGE 2.4.3 - CODE TO ' END K' CHECKS IF CELL FLOODS FROM LEFT
C                   AND CHANGES FLAGS ACCORDINGLY
C
C     ********************************************************************
C
      IF ( FLD(33,3,IFLAG(IK1,J)) .GE. 6 ) GO TO 607
      IF ( (H(IK1,J)-ZZ) .LT. DELTA ) GO TO 607
      IFLAG(IK1,J)=IFLAG(IK1,J) .AND. 504
      IFLAG(IK,J)=IFLAG(IK,J) .AND. 191
      IJM=1
      IF ( IFLAG(IK,J+1) .NE. 502 ) GO TO 606
      IF ( FLD(30,3,IFLAG(IK,J)) .EQ. 7 ) GO TO 606
      FLD(30,3,IFLAG(IK,J))=3
  606 IF ( FLD(33,3,IFLAG(IK,J)) .EQ. 7 ) GO TO 607
      FLD(33,3,IFLAG(IK,J))=3
C
C     ********************************************************************
C
C     END K  --  END OF STAGE 2.4.3  ---  END K
C
C     ********************************************************************
C     ********************************************************************
C
C     STAGE 2.4.4 - CODE TO 'END L' CHECKS IF CELL IS FLOODED FROM
C                   BOTTOM AND CHANGES FLAGS ACCORDINGLY
C
C     ********************************************************************
C
  607 IF ( FLD(30,3,IFLAG(IK,JM1)) .GE. 6 ) GO TO 610
      IF ( (H(IK,JM1)-ZZ) .LT. DELTA ) GO TO 610
      IFLAG(IK,JM1)=IFLAG(IK,JM1) .AND. 455
      IFLAG(IK,J)=IFLAG(IK,J) .AND. 127
      IF ( IJM .EQ. 1 ) GO TO 610
      IJM=1
      IF ( IFLAG(IK,J+1) .NE. 502)  GO TO 608
      IF ( FLD(30,3,IFLAG(IK,J)) .EQ. 7 ) GO TO 608
      FLD(30,3,IFLAG(I,J))=3
  608 IF ( FLD(33,3,IFLAG(IK,J)) .EQ. 7 ) GO TO 610
      FLD(33,3,IFLAG(IK,J))=3
C
C     ********************************************************************
```

TABLE 5.3 - continued

LISTING OF SUBROUTINE MOD

```
C
C     END L  ---  END OF STAGE 2.4.4  ---  END L
C
C     ******************************************************************
C     ******************************************************************
C
C     STAGE 2.4.5 - CODE FROM HERE TO 'END M' CHECKS IC CELL IF
C                   FLOODED FROM TOP AND CHANGES FLAGS ACCORDINGLY
C
C     ******************************************************************
C
  610 IF ( FLD(30,3,IFLAG(IK,J)) .GE. 6 ) GO TO 620
      IF ( (H(IK,JP1)-ZZ) .LT. DELTA ) GO TO 620
      IFLAG(IK,JP1)=IFLAG(IK,JP1) .AND. 383
      IFLAG(IK,J)=IFLAG(IK,J) .AND. 199
      IF ( IJM .EQ. 1 ) GO TO 620
      IJM=1
      IF ( FLD(33,3,IFLAG(IK,J)) .EQ. 7 ) GO TO 618
      FLD(33,3,IFLAG(IK,J))=3
C
C     ******************************************************************
C
C     END M  ---  END OF STAGE 2.4.5  ---  END M
C
C     ******************************************************************
C
  618 IF( ABS(DELTA) .GT. .0001 ) GO TO 640
C
C     ******************************************************************
C     STAGE 2.4.6 - THE CODE TO ' END N' FNDS THE WATER LEVEL IN THE
C                 NEWLY FLOODED CELL.  THE WATER LEVEL IS SET TO THE
C                 AVERAGE WATER LEVEL IN THE ADJACENT CELLS THAT ARE
C                 FLOODED
C
C     ******************************************************************
C
  620 SUM=0.
      IC=0
      IF ( D(IK1 ,J) .LT. 0. ) GO TO 450
      SUM=H(IK1 ,J)
      IC=1
  450 IF ( D(IK,JM1) .LT. 0. ) GO TO 452
      SUM=SUM+H(IK,JM1)
```

TABLE 5.3 - continued

LISTING OF SUBROUTINE MOD

```
      IC=IC+1
  452 IF ( D(IK,JP1) .LT. 0. ) GO TO 453
      SUM=SUM+H(IK,JP1)
      IC=IC+1
  453 SUM=SUM/IC
      H(IK,J)=SUM
C
C     ****************************************************************
C
C     END N  ---  END OF STAGE 2.4.6  ---  END N
C
C     ****************************************************************
C
      D(IK,J)=H(IK,J)-ZZ
      DELTA=DELTAS
      IF ( D(IK,J) .LE. 0. ) GO TO 601
C
C     ****************************************************************
C     STAGE 2.4.7 - CHECK FOR TRACE BACK
C     ****************************************************************
C
      IF ( SUM .GT. Z(IK1,J) .AND. D(IK1,J) .LT. 0. ) GO TO 454
      IF ( SUM .GT. Z(IK,JM1) .AND. D(IK,JM1) .LT. 0. ) GO TO 454
      GO TO 640
  454 CALL TRACE(IK,J,Z)
  640 CONTINUE
C
C     ****************************************************************
C
C      END 0  -- END OF STAGE 2.4  ---  END 0
C
C     ****************************************************************
C
      DELTA = DELTAS
  455 IF ( IBU .EQ. 0 ) GO TO 800
C
C     ****************************************************************
C     STAGE 3.0 - THE LOOP TO 700 CHECKS CELLS IN THE ROW WITH J=JMAX
C     ****************************************************************
C
      DO 700 I=IBU,IK1
      IM1=I-1
      IP1=I+1
```

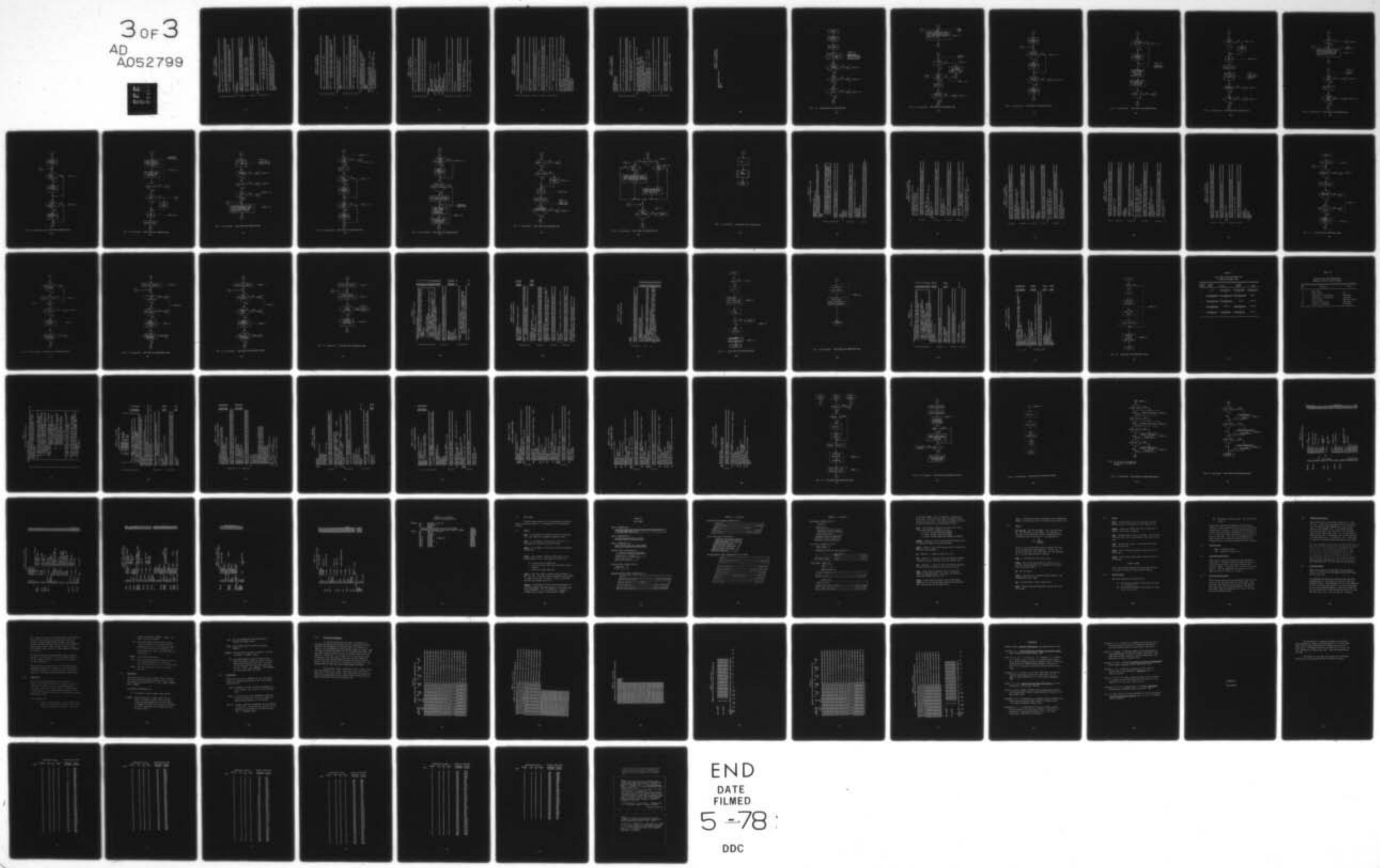
TABLE 5.3 - continued

LISTING OF SUBROUTINE MOD

```
      IGO=0
      IJM=0
C
C     *******************************************************************
C     STAGE 3.1 - CHECK IF CELL IS ALWAYS FLOODED OR ALWAYS DRY
C     *******************************************************************
C
      IF ( FLD(27,1,IFLAG(I,JMAX)) .EQ. 1 ) GO TO 700
C
C     *******************************************************************
C     STAGE 3.2 - IS CELL DRY - IF YES JUMP TO SEE IF IT FLOODS
C     *******************************************************************
C
      IF ( IFLAG(I,JMAX) .GE. 237 ) GO TO 650
C
C     *******************************************************************
C
C     STAGE 3.3 - THE CODE FROM HERE TO 'END P' CHECKS IF THE CELL
C                 DRIES UP AND CHANGES FLAGS ACCORDINGLY
C
C     *******************************************************************
C
      IF ( D(I,JMAX) .GT. DELTA ) GO TO 700
  641 CONTINUE
      IF ( FLD(30,3,IFLAG(I,JMAX)).EQ. 3 ) IFLAG(I,JMAX)=IFLAG(I,JMAX)
     * .AND. 455
      IF ( FLD(33,3,IFLAG(I,JK)) .EQ. 3)IFLAG(I,JK)=IFLAG(I,JK).AND. 504
      IFLAG(I,JMAX) = IFLAG(I,JMAX) .OR. MASK
      IFLAG(IM1,JMAX)=IFLAG(IM1,JMAX) .OR. 5
      IFLAG(I,JK1)=IFLAG(I,JK1) .OR. 40
      IFLAG(IP1,JMAX)=IFLAG(IP1,JMAX) .OR. 64
C
C     *******************************************************************
C     STAGE 3.3.1 - SET APPROPRIATE FLOWS TO ZERO (0) AND DEPTHS AND
C                   WATER LEVELS TO -1000.
C     *******************************************************************
C
      QX(I,IMAX)=0.
      QY(I,JMAX)=0.
      QX(IM1,JMAX)=0.
      QY(I,JK1)=0.
      D(I,JMAX)=-1000.
      H(I,JMAX)=-1000.
```

AD-A052 799 TRACOR INC AUSTIN TEX F/G 8/8
COMPARISON OF NUMERICAL AND PHYSICAL HYDRAULIC MODELS, MASONBOR--ETC(U)
JUN 77 R J CHEN, L A HEMBREE DACW72-72-C-0029
UNCLASSIFIED CERC-GITI-6-APP-3 NL

3 OF 3
AD
A052799



END
DATE
FILMED
5-78
DDC

TABLE 5.3 - continued

LISTING OF SUBROUTINE MOD

```
      GO TO 700
C     ******************************************************************
C
C     END P  ---  END OF STAGE 3.0  ---  END P
C
C     ******************************************************************
C     ******************************************************************
C
C     STAGE 3.4 - STAGE FROM HERE TO END V CHECKS IF CELL FLOODS
C                 AND CHANGES FLAGS AND DEPTHS ACCORDINGLY
C
C     ******************************************************************
C
  650 CONTINUE
      ZZ=Z(I,JMAX)
C
C     ******************************************************************
C     STAGE 3.4.1 - CHECK IF CELL FLOODS FROM ANY DIRECTION
C     ******************************************************************
C
      IF((H(IM1,JK)-ZZ) .GT. DELTA ) GO TO 661
      IF ( (H(IP1,JK)-ZZ) .GT. DELTA ) GO TO 661
      IF ( (H(I,JK1)-ZZ) .GT. DELTA ) GO TO 661
C
C     ******************************************************************
C     STAGE 3.4.2 - CELL DOES NOT FLOOD -- JUMP TO END OF LOOP
C     ******************************************************************
C
      GO TO 700
  661 DELTA=0.
C
C     ******************************************************************
C
C     STAGE 3.4.3 - CODE FROM HERE TO 'END R' CHECKS IF CELL FLOODS
C                   FROM LEFT AND CHANGES FLAGS ACCORDINGLY
C
C     ******************************************************************
C
      IF( FLD(33,3,IFLAG(IM1,JMAX)) .GE. 6 ) GO TO 660
      IF ( (H(IM1,JMAX)-ZZ) .LT. DELTA ) GO TO 660
      IFLAG(IM1,JMAX)=IFLAG(IM1,JMAX) .AND. 504
      IFLAG(I,JMAX)=IFLAG(I,JMAX) .AND. 191
      IJM=1
```

TABLE 5.3 - continued

LISTING OF SUBROUTINE MOD

```
C
C     ******************************************************************
C
C     END R  ---  END OF STAGE 3.4.3  ---  END R
C
C     ******************************************************************
C     ******************************************************************
C
C     STAGE 3.4.4 - CODE FROM HERE TO 'END S' CHECKS IF CELL FLOODS
C                   FROM BOTTOM AND CHANGES FLAGS ACCORDINGLY
C
C     ******************************************************************
C
  660 IF ( (H(I,JK1)-ZZ) .LT. DELTA ) GO TO 670
      IFLAG(I,JK1)=IFLAG(I,JK1) .AND. 455
      IFLAG(I,JMAX) =IFLAG(I,JMAX) .AND. 127
      IJM=1
C
C     ******************************************************************
C
C     END S  ---  END OF STAGE 3.4.4  ---  END S
C
C     ******************************************************************
C     ******************************************************************
C
C     STAGE 3.4.5 - CODE FROM HERE TO 'END T' CHECKS IF CELL FLOODS
C                   FROM RIGHT AND CHANGES FLAGS ACCORDINGLY
C
C     ******************************************************************
C
  670 IF ( FLD(33,3,IFLAG(I,JMAX)) .GE. 6 ) GO TO 690
      IF ( (H(IP1,JMAX)-ZZ) .LT. DELTA ) GO TO 690
      IFLAG(IP1,JMAX)=IFLAG(IP1,JMAX) .AND. 447
      IFLAG(I,JMAX)=IFLAG(I,JMAX) .AND. 248
      IGO=1
      IJM=1
  690 IF ( IJM .NE. 1 ) GO TO 700
  691 IF ( FLD(30,3,IFLAG(I,JMAX)) .EQ. 7 ) GO TO 701
      FLD(30,3,IFLAG(I,JMAX))=3
  701 IF ( IFLAG(IP1,JK) .NE. 502 ) GO TO 692
      IF ( FLD(33,3,IFLAG(I,JK)) .EQ. 7 ) GO TO 692
      FLD(33,3,IFLAG(I,JK))=3
C
```

TABLE 5.3 - continued

LISTING OF SUBROUTINE MOD

```
C      ******************************************************************
C
C      END T  ---  END OF STAGE 3.4.5  ---  END T
C
C      ******************************************************************
C      ******************************************************************
C
C      STAGE 3.4.6 - CODE FROM HERE TO 'END U' FINDS NEW WATER LEVELS
C
C      ******************************************************************
C
  692 IC=0
      SUM=0.
       IF ( D(IM1,JMAX) .LT. 0 ) GO TO 470
      SUM=H(IM1,JMAX)
      IC=1
  470 IF ( D(I,JK1) .LT. 0.)  GO TO 471
      SUM=SUM+H(I,JK1)
      IC=IC+1
  471 IF ( D(IP1,JMAX) .LT. 0. ) GO TO 473
      SUM=SUM+H(IP1,JMAX)
      IC=IC+1
  473 H(I,JMAX)=SUM/IC
      D(I,JMAX)=H(I,JMAX)-ZZ
C
C      ******************************************************************
C
C      END U  --- END OF STAGE 3.4.6  --  END U
C
C      ******************************************************************
C
      DELTA=DELTAS
C
C      ******************************************************************
C      STAGE 3.4.7 - CHECK FOR TRACE BACK
C      ******************************************************************
C
      IF ( D(I,JK) .LE. 0. ) GO TO 641
      IF ( IGO .EQ. 1 ) CALL TRACE( I,JMAX,Z)
C
C      ******************************************************************
C
C      END V  --  END OF STAGE 3.4  ---  END V
```

TABLE 5.3 - continued

LISTING OF SUBROUTINE MOD

```
C
C     ******************************************************************
C     ******************************************************************
C
  700 CONTINUE
C
C     ******************************************************************
C
C     STAGE 4.0 - CHECK CELL (IMAX,JMAX)
C
C     ******************************************************************
      DELTA = DELTAS
C
C     ******************************************************************
C     STAGE 4.1 - IS IT ALWAYS FLOODED OR ALWAYS DRY
C     ******************************************************************
C
      IF ( FLD(27,1,IFLAG(IMAX,JMAX)) .EQ. 1 ) GO TO 800
C
C     ******************************************************************
C     STAGE 4.2 - IS IT DRY - IF YES JUMP TO SEE IF IT FLOODS
C     ******************************************************************
C
      IF ( IFLAG(IMAX,JMAX) .GE. 237 ) GO TO 750
C
C     ******************************************************************
C
C     STAGE 4.3 - CODE FORM HERE TO 'END X' CHECKS IF CELL DRIES UP AND
C                 CHANGES FLAGS ACCORDINFLY
C
C
C     ******************************************************************
C
      IF ( D(IMAX,JMAX) .GT. DELTA ) GO TO 800
      IF(FLD(33,3,IFLAG(IMAX,JMAX)).EQ. 3)IFLAG(IMAX,JMAX)=IFLAG(IMAX,JMAX
     1AX) .AND. 504
      IF (FLD(30,3,IFLAG(IMAX,JMAX)).EQ. 3)IFLAG(IMAX,JMAX)=IFLAG(IMAX,
     1JMAX) .AND. 455
      IFLAG(IMAX,JMAX)=IFLAG(IMAX,JMAX) .OR. MASK
      IFLAG(IK1,JMAX)=IFLAG(IK1,JMAX) .OR. 5
      IFLAG(IMAX,JK1)=IFLAG(IMAX,JK1) .OR. 40
      QX(IMAX,JMAX)=0.
      QY(IMAX,JMAX)=0.
      QX(IK1,JMAX)=0.
```

TABLE 5.3 - continued

LISTING OF SUBROUTINE MOD

```
      QY(IMAX,JK1)=0.
      D(IMAX,JMAX)=-1000.
      H(IMAX,JMAX)=-1000.
      GO TO 800
C
C     ******************************************************************
C
C     END X  ---  END OF STAGE 4.3  ---  END X
C
C     ******************************************************************
C     ******************************************************************
C
C     STAGE 4.4 - CODE FROM HERE TO END Y CHECKS IF CELL FLOODS AND
C                 CHANGES FLAGS ACCORDINGLY
C
C     ******************************************************************
C
  750 IF ( (H(IK1,JMAX)-Z(IMAX,JMAX)) .LT. DELTA ) GO TO 800
      IFLAG(IK1,JMAX)=IFLAG(IK1,JMAX) .AND. 248
      IFLAG(IMAX,JK1)=IFLAG(IMAX,JK1) .AND. 455
      IFLAG(IMAX,JMAX)=IFLAG(IMAX,JMAX) .AND. 191
      IFLAG(IMAX,JMAX)=IFLAG(IMAX,JMAX) .AND. 63
      D(IMAX,JMAX)=H(IK1,JMAX)-Z(IMAX,JMAX)
      H(IMAX,JMAX)=H(IK1,JMAX)
      IF ( FLD(33,3,IFLAG(IMAX,JMAX)) .EQ. 7 ) GO TO 760
      FLD(33,3,IFLAG(IMAX,JMAX))=3
  760 IF ( FLD(30,3,IFLAG(IMAX,JMAX)) .EQ. 7 ) GO TO 800
      FLD(30,3,IFLAG(IMAX,JMAX))=3
C
C     ******************************************************************
C
C     END Y  ---  END OF STAGE 4.4  ---  END Y
C
C     ******************************************************************
C
  800 CONTINUE
C
C     ******************************************************************
C     STAGE 5.0 - CHECK FOR CALL TO MODPRI
C     ******************************************************************
C
      IF ( ITR .NE. 0 .AND .ITR .LT. IPRMOD ) RETURN
      ITR=0
```

TABLE 5.3 - continued

LISTING OF SUBROUTINE MOD

```
      CALL MODPRI(IOP,TIME)
      RETURN
      END
```

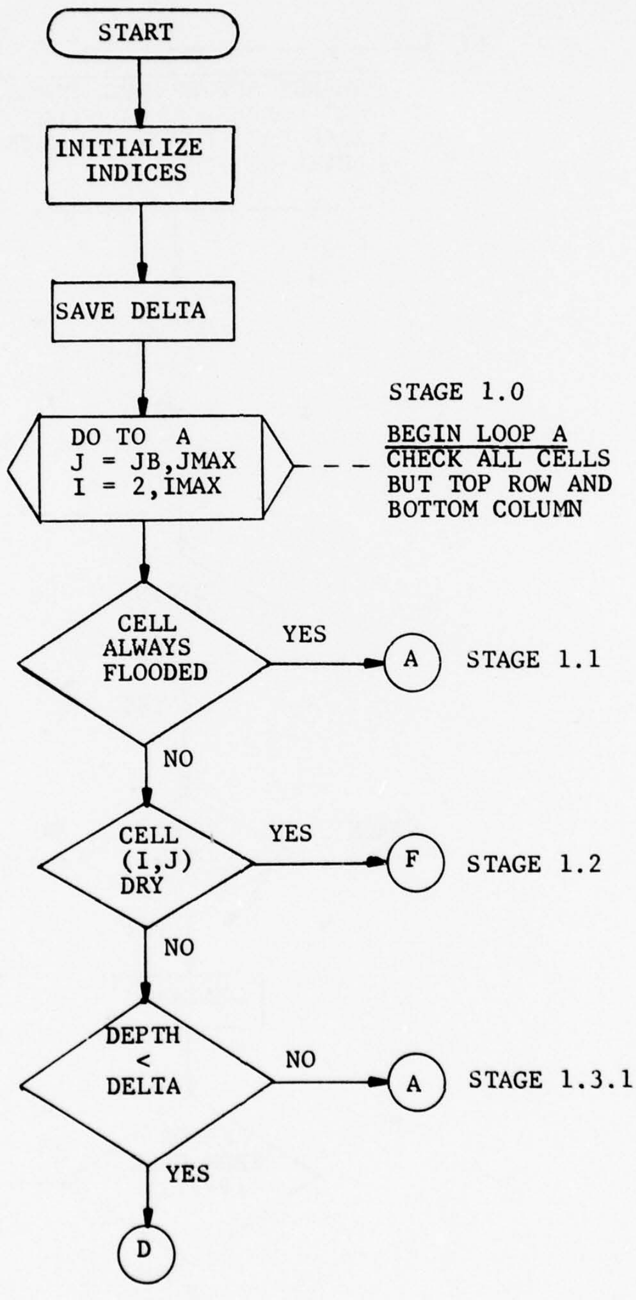


FIG. 5.3 FLOW CHART FOR SUBROUTINE MOD

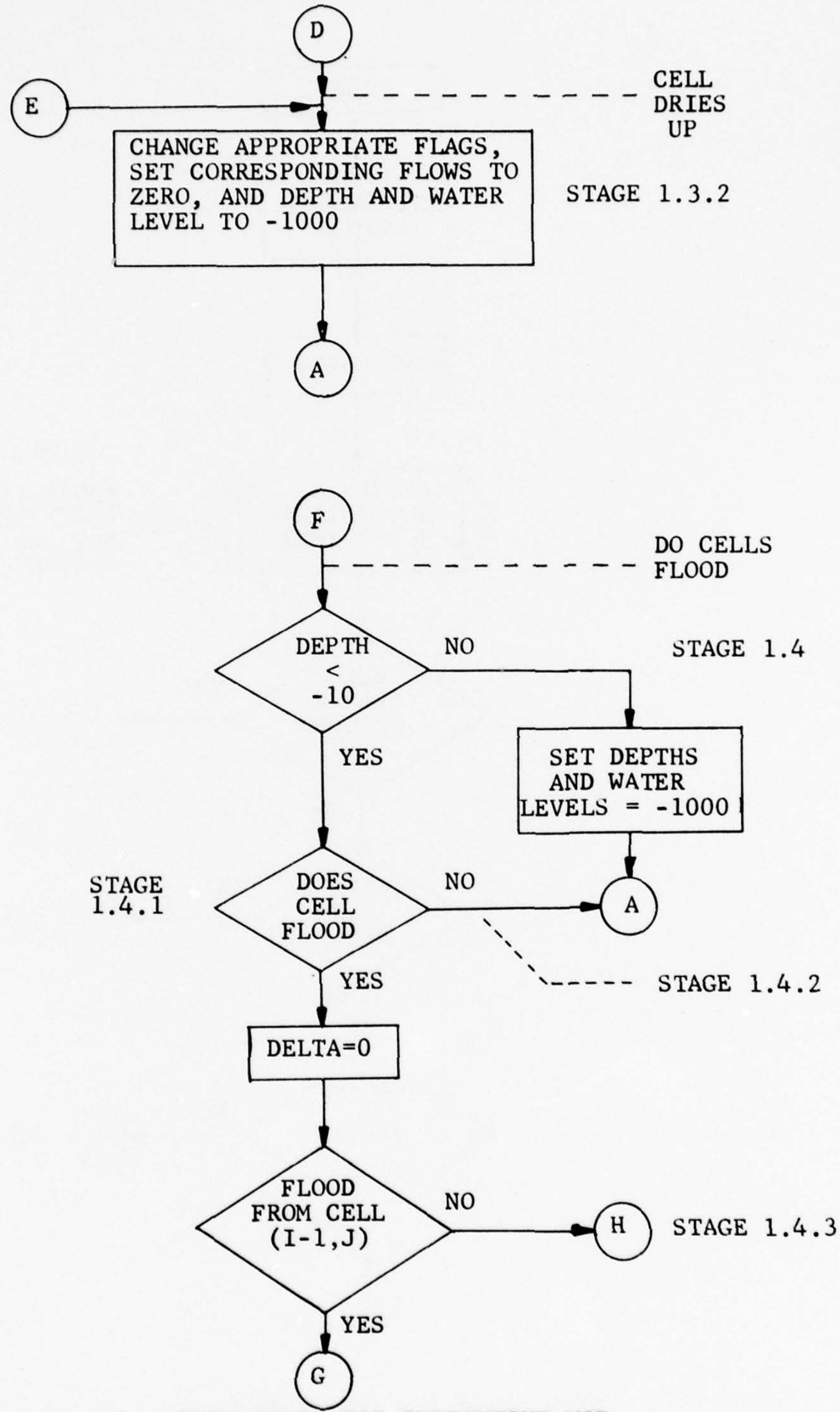


FIG. 5.3 (Continued) FLOW CHART FOR SUBROUTINE MOD

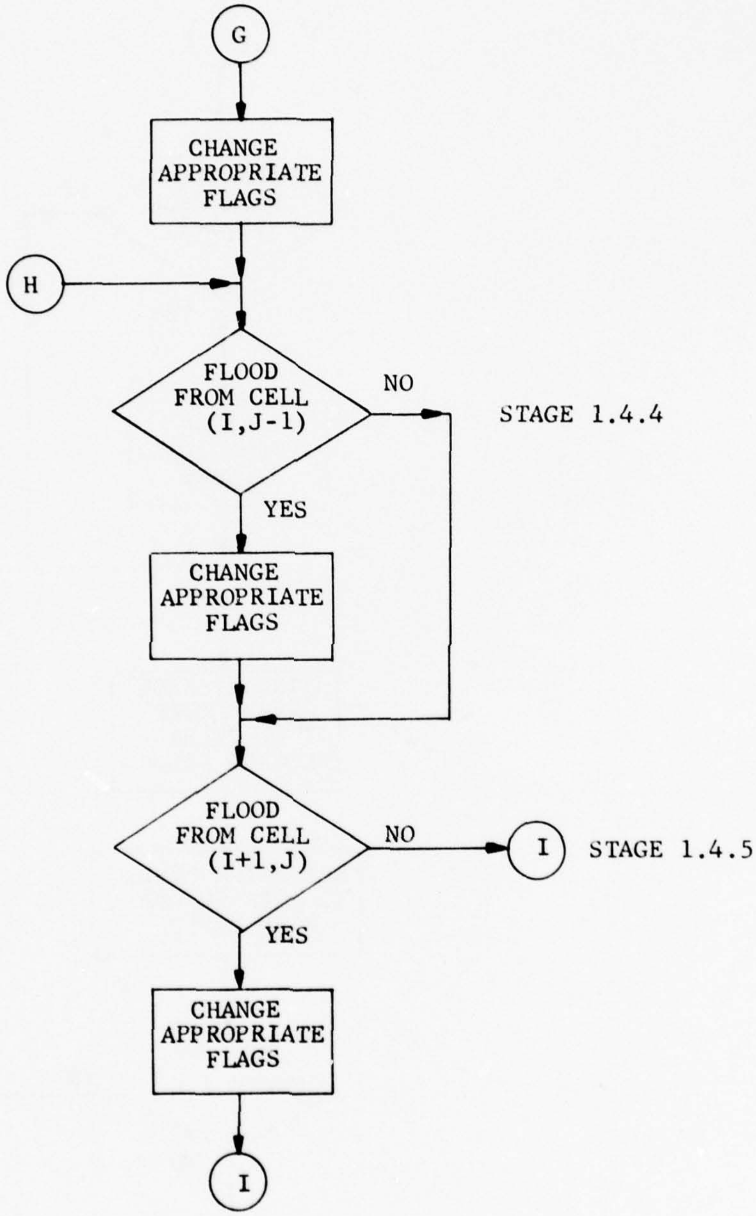


FIG. 5.3 (Continued) FLOW CHART FOR SUBROUTINE MOD

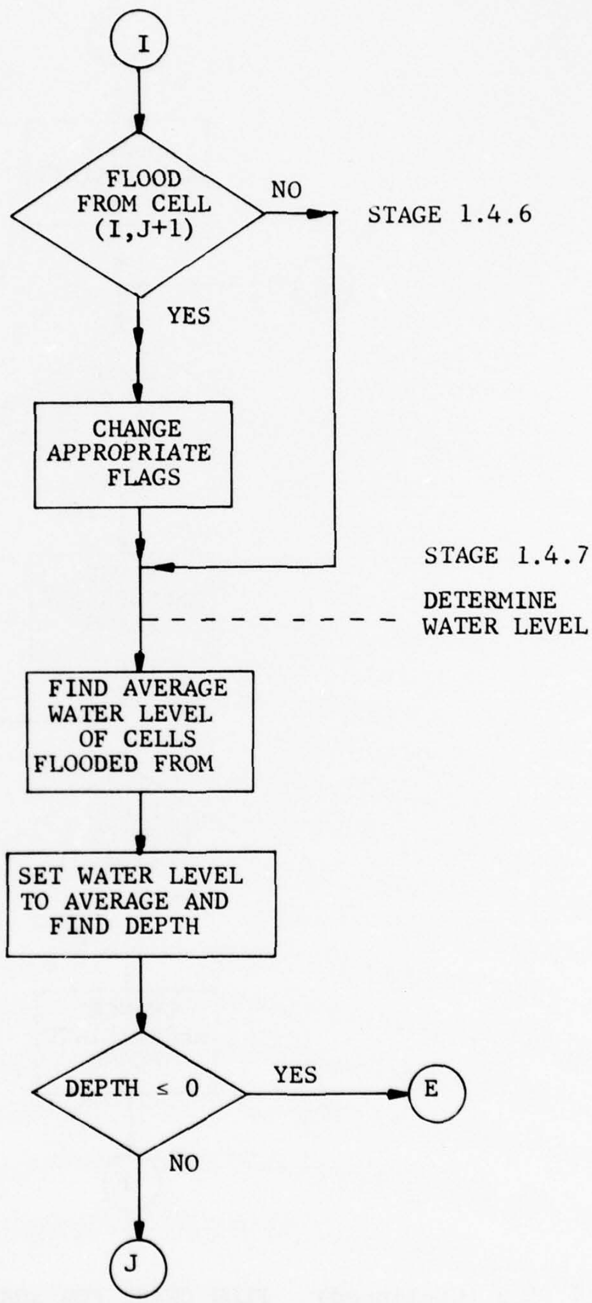


FIG. 5.3 (Continued) FLOW CHART FOR SUBROUTINE MOD

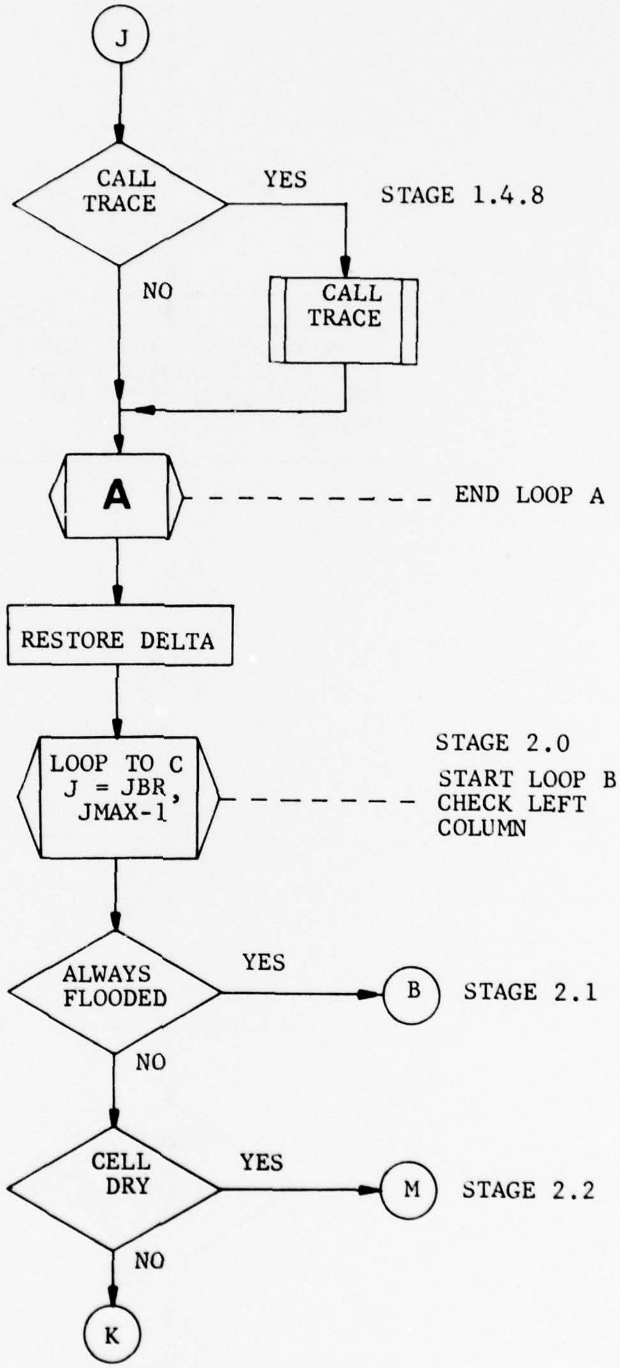


FIG. 5.3 (Continued) FLOW CHART FOR SUBROUTINE MOD

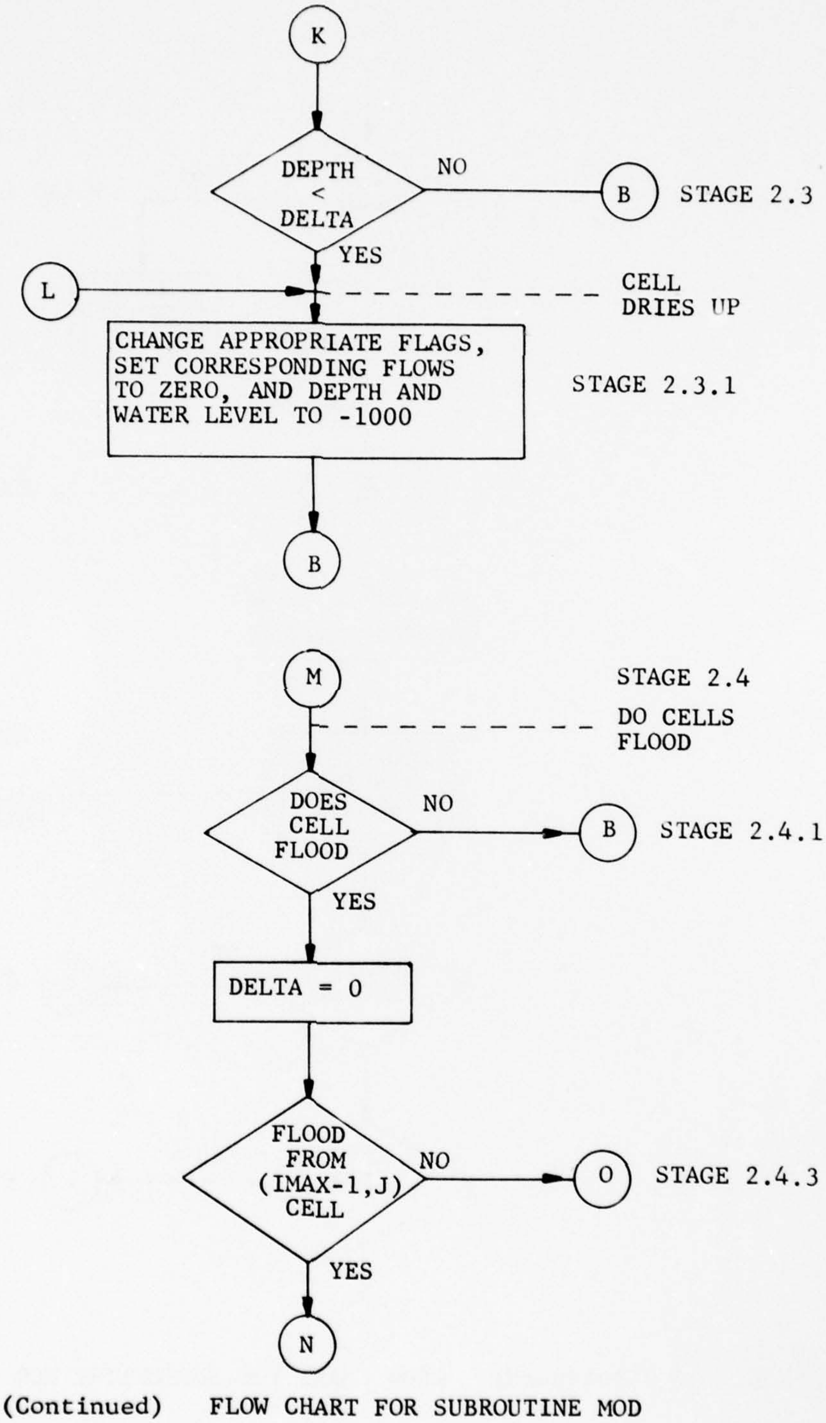


FIG. 5.3 (Continued) FLOW CHART FOR SUBROUTINE MOD

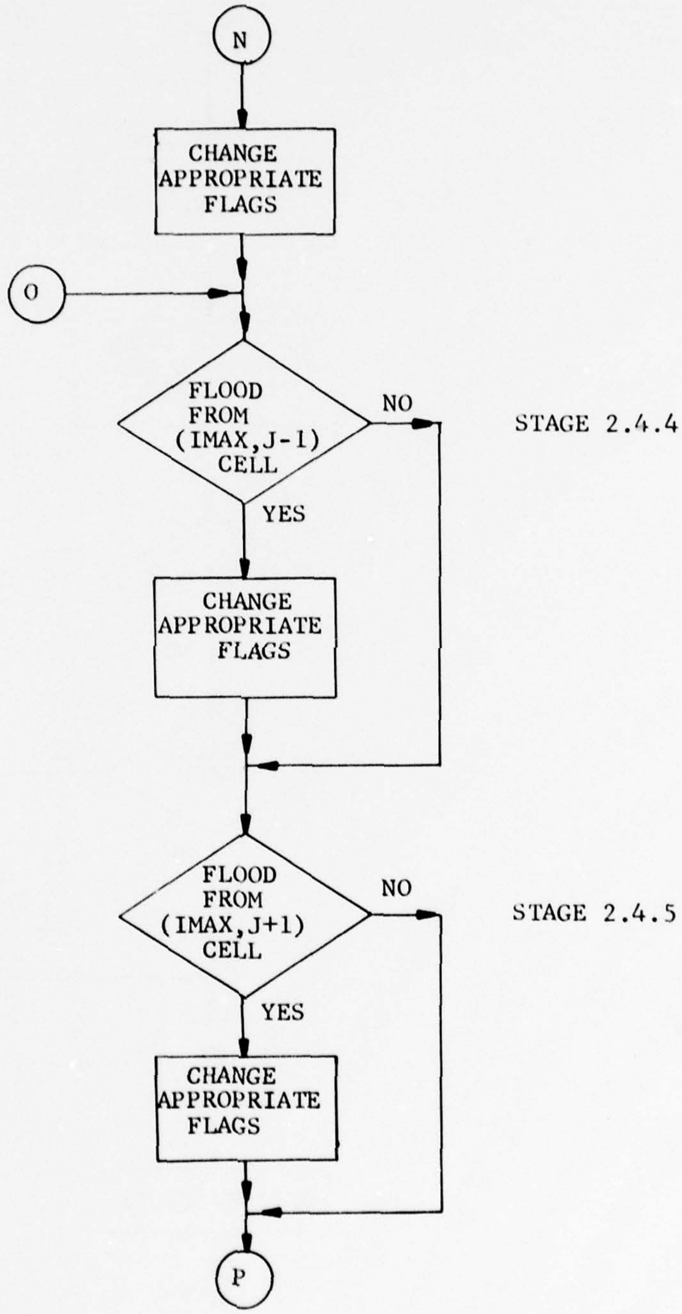


FIG. 5.3 (Continued) FLOW CHART FOR SUBROUTINE MOD

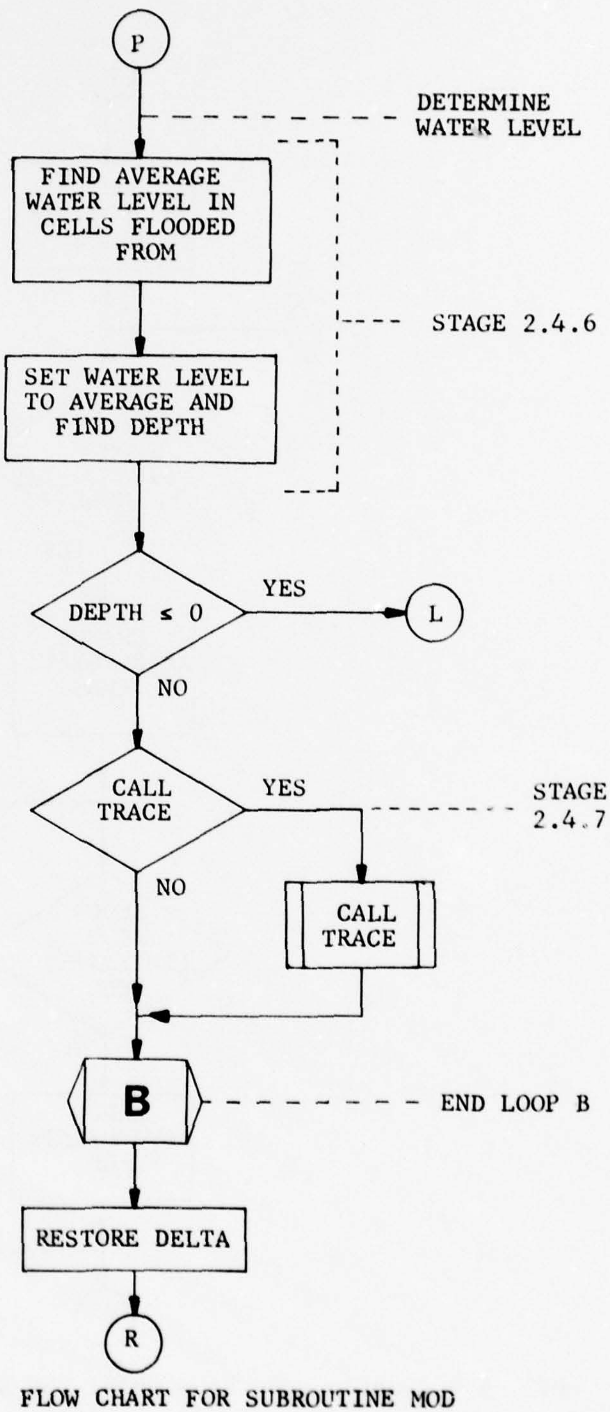


FIG. 5.3 (Continued) FLOW CHART FOR SUBROUTINE MOD

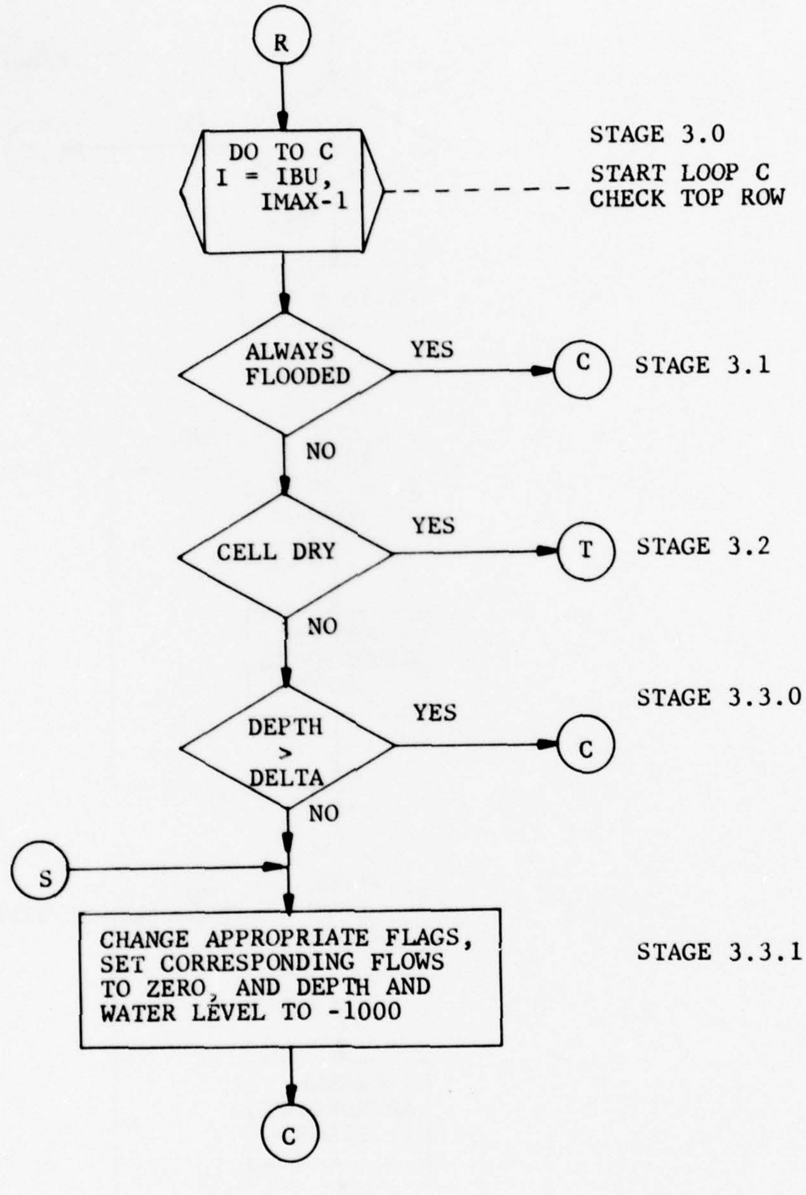


FIG. 5.3 (Continued) FLOW CHART FOR SUBROUTINE MOD

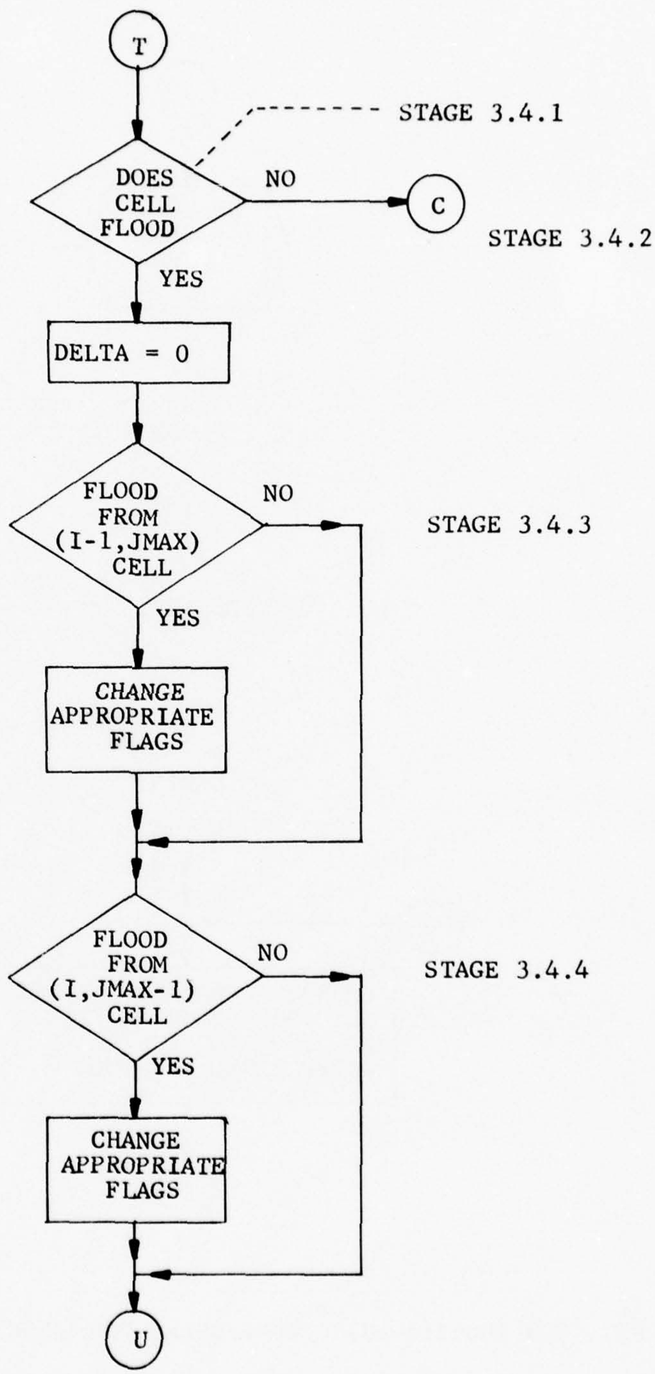


FIG. 5.3 (Continued) FLOW CHART FOR SUBROUTINE MOD

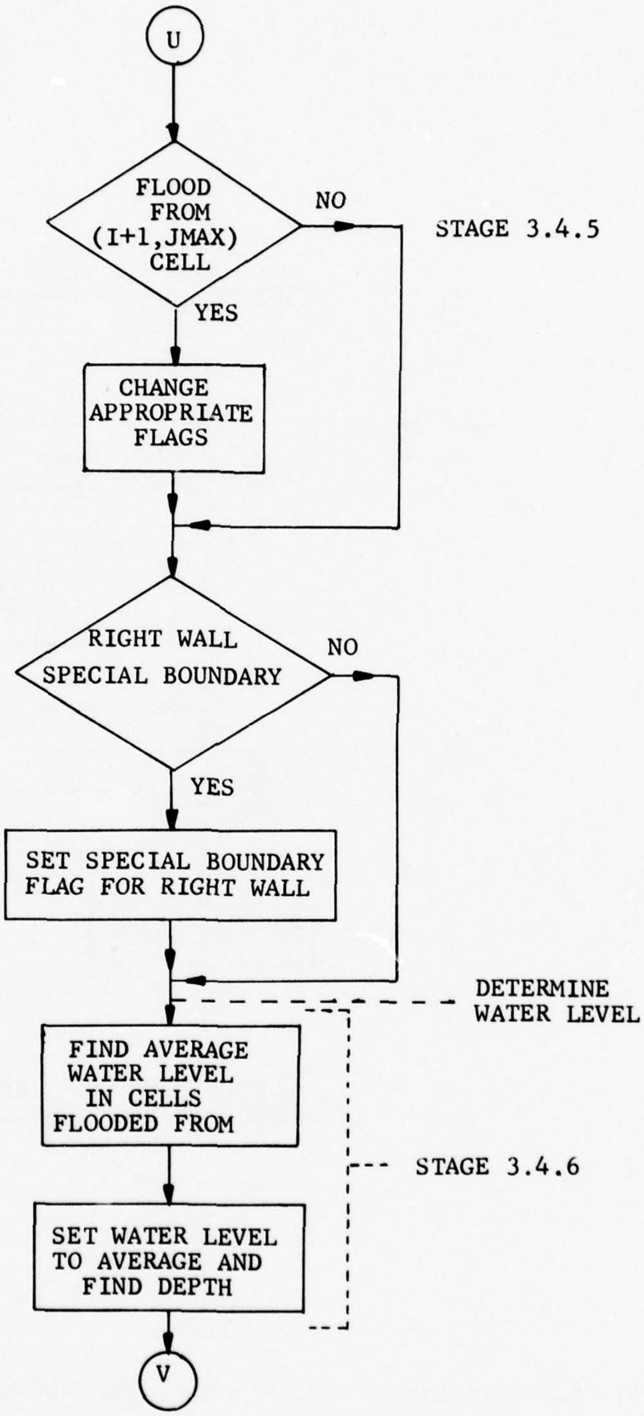


FIG. 5.3 (Continued) FLOW CHART FOR SUBROUTINE MOD

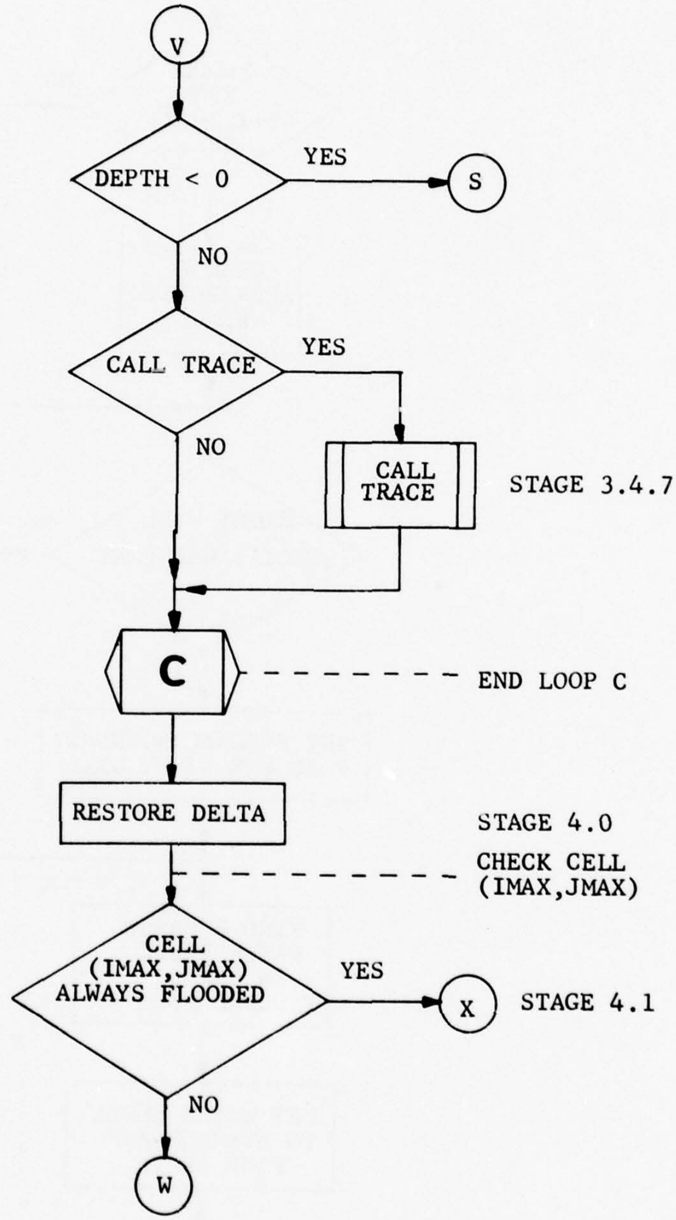


FIG. 5.3 (Continued) FLOW CHART FOR SUBROUTINE MOD

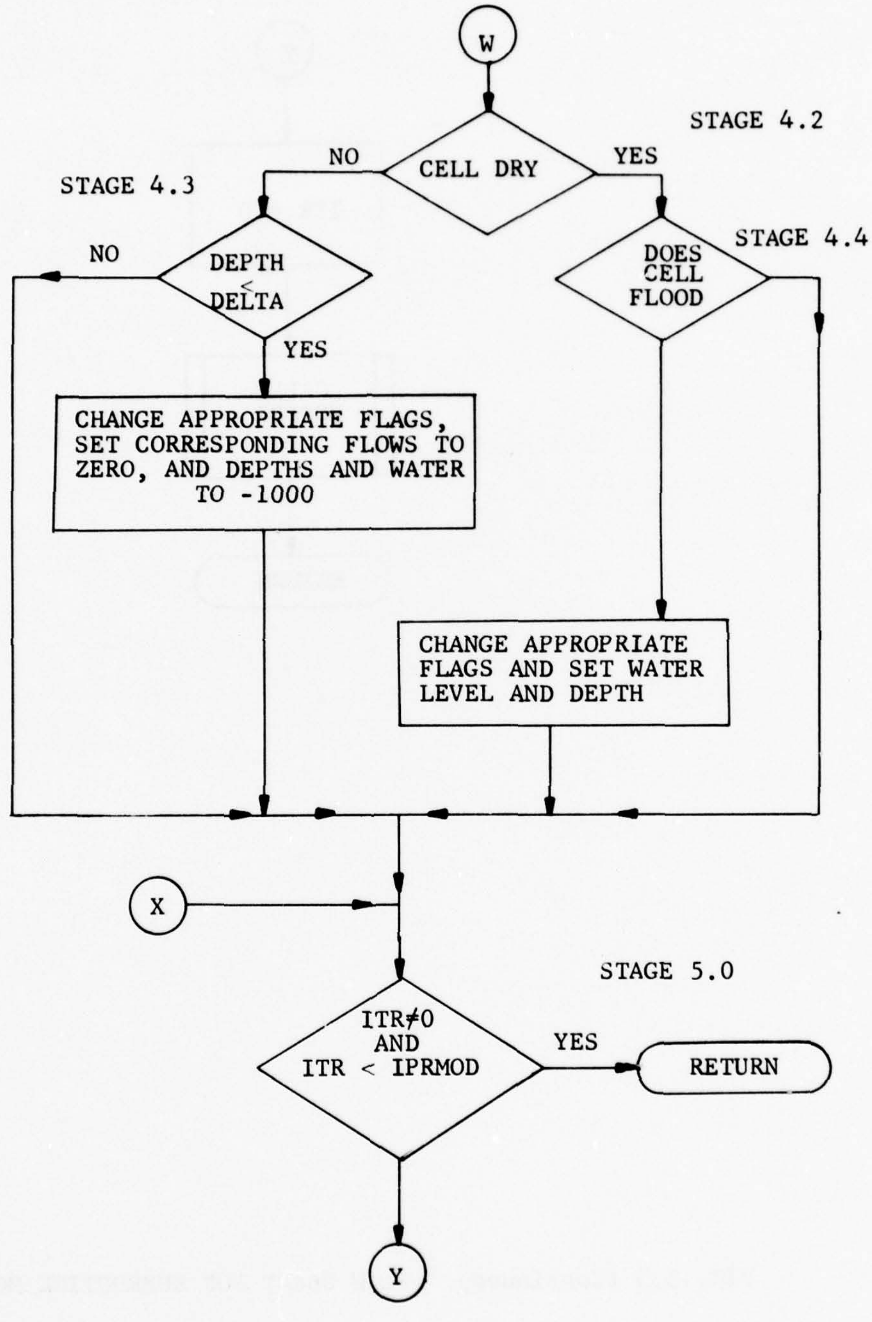


FIG. 5.3 (Continued) FLOW CHART FOR SUBROUTINE MOD

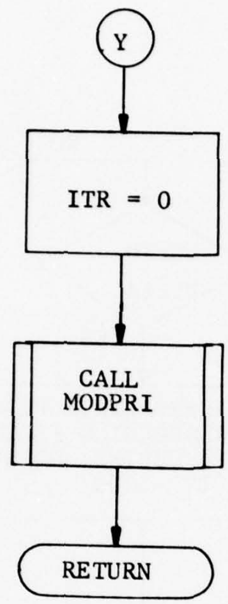


FIG. 5.3 (Continued) FLOW CHART FOR SUBROUTINE MOD

TABLE 5.4

LISTING OF SUBROUTINE TRACE

```
      SUBROUTINE TRACE(I,J,Z)
      DIMENSION IS(50),JS(50),Z(30,25)
      COMMON/A/QX(30,25),QY(30,25),D(30,25),IFLAG(30,25),H(30,25),IMAX,
     1     JMAX,DELTA,JB,JBR,IBU,PUNV(5,4,50),NPU,PCHT(50),IPCH
      COMMON /B/DUP(30,25),DX,DY
      MS=0
      NS=0
C
C         ****************************************************************
C         * THIS SUBROUTINE CHECKS IF ANY OF THE CELLS ALREADY CHECKED  *
C         * BY SUBROUTINE MOD ARE FLOODED FROM THE CELL JUST FLOODED    *
C         * BY MOD OR A CELL FLOODED BY A CALL     TO THIS SUBROUTINE   *
C         ****************************************************************
C
C
C     ********************************************************************
C     STAGE 1.0 - SAVE INDICES I AND J
C     ********************************************************************
C
      II=I
      JJ=J
   10 IP1=II+1
      IM1=II-1
      JP1=JJ+1
      JM1=JJ-1
C
C     ********************************************************************
C     STAGE 2.0 - CHECK FOR SPECIAL BOUNDARY ON RIGHT WALL
C     ********************************************************************
C
      IF ( FLD(33,3,IFLAG(II,JJ)) .EQ. 3 ) GO TO 51
      HI=H(II,JJ)
      HID=HI-DELTA
      ZZ=Z(II,JJ)
      IFL=IFLAG(II,JJ)
C
C     ********************************************************************
C     STAGE 3.0 - CHECK IF CELL TO LEFT (I-1,J) FLOODS OR IS ALREADY FLOODED
C     ********************************************************************
```

TABLE 5.4 - continued

LISTING OF SUBROUTINE TRACE

```
C
      IF ( FLD(29,1,IFL) .EQ. 0 ) GO TO 20
      IFLM=FLD(33,3,IFLAG(IM1,JJ))
      IF ( IFLM .GE. 6 ) GO TO 20
      IF ( IFLM .EQ. 3 ) GO TO 15
      IF ( Z(IM1,JJ) .GE. HID) GO TO 20
C
C     ******************************************************************
C      STAGE 3.1 - CELL (I-1,J) FLOODS - SET APPOPRIATE FLAGS
C     ******************************************************************
C
      IFL=IFL .AND. 191
      IF ( IM1 .EQ. 1 ) GO TO 140
      IFLAG(IM1,JJ)=IFLAG(IM1,JJ) .AND. 248
      GO TO 140
   15 IFL=IFL .AND. 191
  140 CONTINUE
C
C     ******************************************************************
C      STAGE 3.2 - SET D(I-1,J) AND H(I-1,J) TO CORRECT VALUES
C     ******************************************************************
C
      IF ( DUP(IM1,JJ) .NE. 0 ) GO TO 20
      DUP(IM1,JJ)=1.0
      D(IM1,JJ)=HI-Z(IM1,JJ)
      H(IM1,JJ)=HI
C
C     ******************************************************************
C     STAGE 3.3 - STORE INDICES (I-1,J) FOR FURTHER TRACE BACK
C     ******************************************************************
C
      NS=NS+1
      IS(NS)=IM1
      JS(NS)=JJ
C
C     ******************************************************************
C     STAGE 4.0 - CHECK IF BOTTOM CELL (I,J-1) FLOODS  OR IS
C                 ALREADY FLOODED
C     ******************************************************************
C
   20 IF ( Z(II,JM1) .GE. HID) GO TO 30
      IF ( FLD(30,3,IFLAG(II,JM1)) .GE. 6 ) GO TO 30
      IF ( FLD(28,1,IFL ) .EQ.  0 ) GO TO 30
```

TABLE 5.4 - continued

LISTING OF SUBROUTINE TRACE

```
C
C     ********************************************************************
C     STAGE 4.1 - CELL (I,J-1) FLOODS - SET APPROPRIATE FLAGS
C     ********************************************************************
C
      IFL=IFL .AND. 127
      IFLAG(II,JM1)=IFLAG(II,JM1) .AND. 455
      IF ( DUP(II,JM1) .NE. 0 ) GO TO 30
      DUP(II,JM1)=1.0
C
C     ********************************************************************
C     STAGE 4.2 - SET D(I,J-1) AND H(I,J-1) TO CORRECT VALUES
C     ********************************************************************
C
      D(II,JM1)=HI-Z(II,JM1)
      H(II,JM1)=HI
C
C     ********************************************************************
C     STAGE 4.3 - STORE INDICES I AND J-1 FOR FURTHER TRACE BACK
C     ********************************************************************
C
      NS=NS+1
      IS(NS)=II
      JS(NS)=JM1
C
C     ********************************************************************
C     STAGE 5.0 - CHECK IF TOP CELL (I,J+1) FLOODS OR IS ALREADY FLOODED
C     ********************************************************************
C
   30 IF ( JJ .EQ. J ) GO TO 40
      IF ( FLD(30,3,IFL) .EQ. 0 ) GO TO 40
      IF ( FLD(30,3,IFL) .GE. 6 ) GO TO 40
      IF ( Z(II,JP1) .GE. HID) GO TO 40
C
C     ********************************************************************
C     STAGE 5.1 - CELL (I,J+1) FLOODS - SET APPROPRIATE FLAGS
C     ********************************************************************
C
      IFL=IFL .AND. 199
      IFLAG(II,JP1)=IFLAG(II,JP1) .AND. 127
      IF ( DUP(II,JP1) .NE. 0 ) GO TO 40
      DUP(II,JP1)=1.
C
```

TABLE 5.4 - continued

LISTING OF SUBROUTINE TRACE

```
C     ******************************************************************
C     STAGE 5.2 - SET D(I,J+1) AND H(I,J+1) TO CORRECT VALUES
C     ******************************************************************
C
      D(II,JP1)=HI-Z(II,JP1)
      H(II,JP1)=HI
C
C     ******************************************************************
C     STAGE 5.3 - STORE INDICES I AND J+1 FOR FURTHER TRACE BACK
C     ******************************************************************
C
      NS=NS+1
      IS(NS)=II
      JS(NS)=JP1
C
C     ******************************************************************
C     STAGE 6.0 - CHECK IF CELL TO RIGHT (I+1,J) FLOODS OR IS ALREADY
C                 FLOODED
C     ******************************************************************
C
   40 IF ( FLD(33,3,IFL) .EQ. 0 ) GO TO 50
      IF ( FLD(33,3,IFL) .GE. 6 ) GO TO 50
      IF ( Z(IP1,JJ) .GT. ZZ ) GO TO 50
      IF ( Z(IP1,JJ) .GE. HID) GO TO 50
      IF ( JJ .EQ. J .AND. II .EQ. I ) GO TO 50
C
C     ******************************************************************
C     STAGE 6.1 - CELL (I+1,J) FLOODS - SET APPROPRIATE FLAGS
C     ******************************************************************
C
      IFL=IFL .AND. 248
      IFLAG(IP1,JJ)=IFLAG(IP1,JJ) .AND. 447
      IF ( DUP(IP1,JJ) .NE. 0 ) GO TO 50
      DUP(IP1,JJ)=1.0
      NS=NS+1
C
C     ******************************************************************
C     STAGE 6.2 - SET D(I+1,J) AND H(I+1,J) TO CORRECT VALUES
C     ******************************************************************
C
      D(IP1,JJ)=HI-Z(IP1,JJ)
      H(IP1,JJ)=HI
C
```

TABLE 5.4 - continued

LISTING OF SUBROUTINE TRACE

```
C     *********************************************************************
C     STAGE 6.3 - STORE INDICES I+1 AND J FOR FURTHER TRACE BACK
C     *********************************************************************
C
      IS(NS)=IP1
      JS(NS)=JJ
C
C     *********************************************************************
C     STAGE 7.0 - STORE NEW FLAG FOR CELL (II,JJ)
C     *********************************************************************
C
   50 IFLAG(II,JJ)=IFL
   51 MS=MS+1
C
C     *********************************************************************
C     STAGE 8.0 - CHECK IF ALL CELLS HAVE BEEN CHECKED
C     *********************************************************************
C
      IF ( MS .GT. NS ) GO TO 60
C
C     *********************************************************************
C     STAGE 9.0 - RETEIVE NEW INDICES FOR NEXT CELL
C     *********************************************************************
C
      II=IS(MS)
      JJ=JS(MS)
      DUP(II,JJ)=0.
      GO TO 10
   60 CONTINUE
      RETURN
      END
```

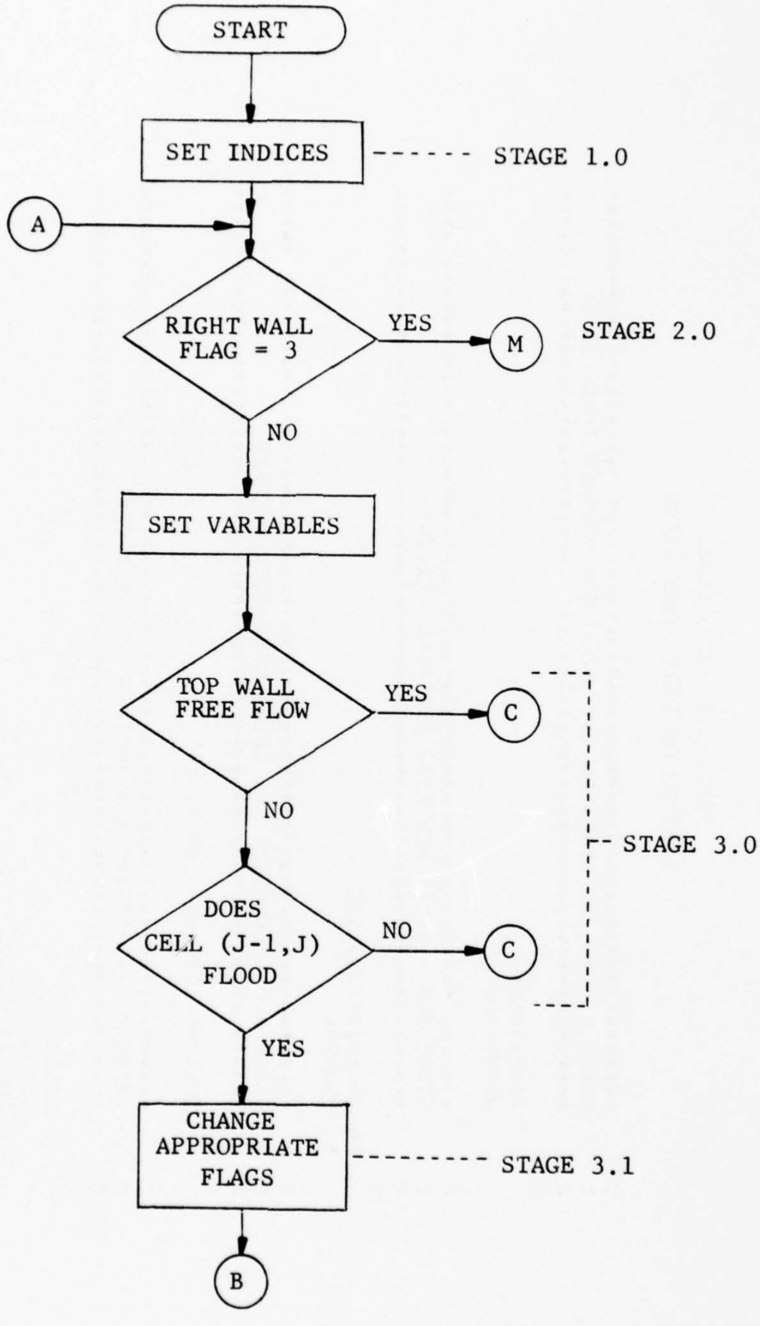


FIG. 5.4 FLOW CHART FOR SUBROUTINE TRACE

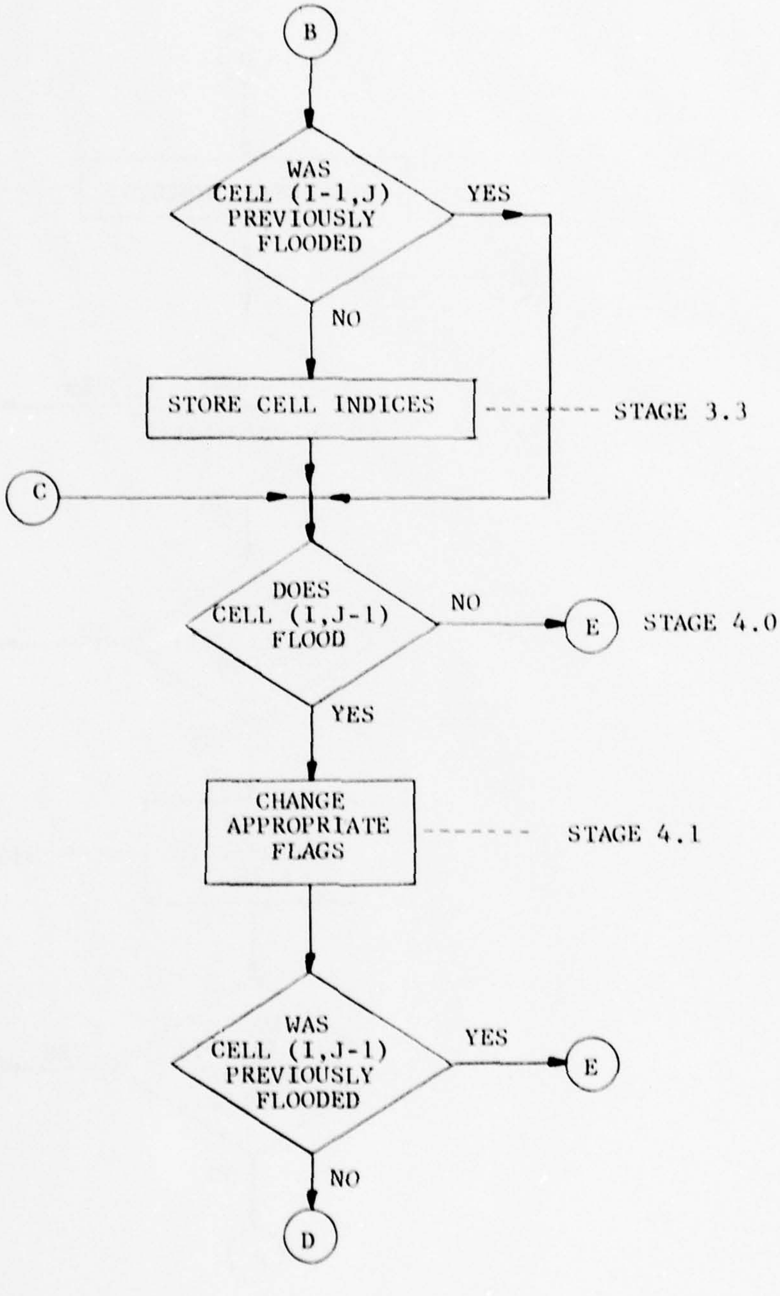


FIG. 5.4 (Continued) FLOW CHART FOR SUBROUTINE TRACE

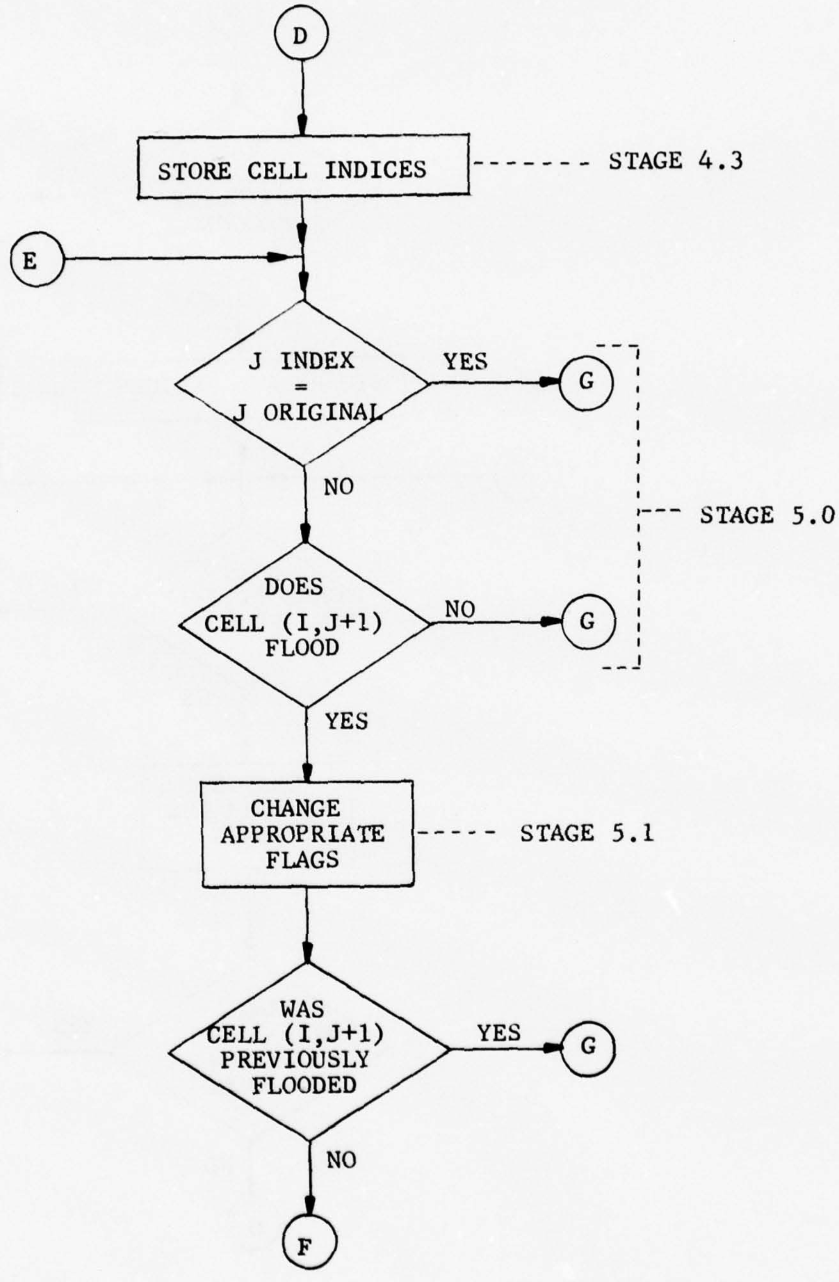


FIG. 5.4 (Continued) FLOW CHART FOR SUBROUTINE TRACE

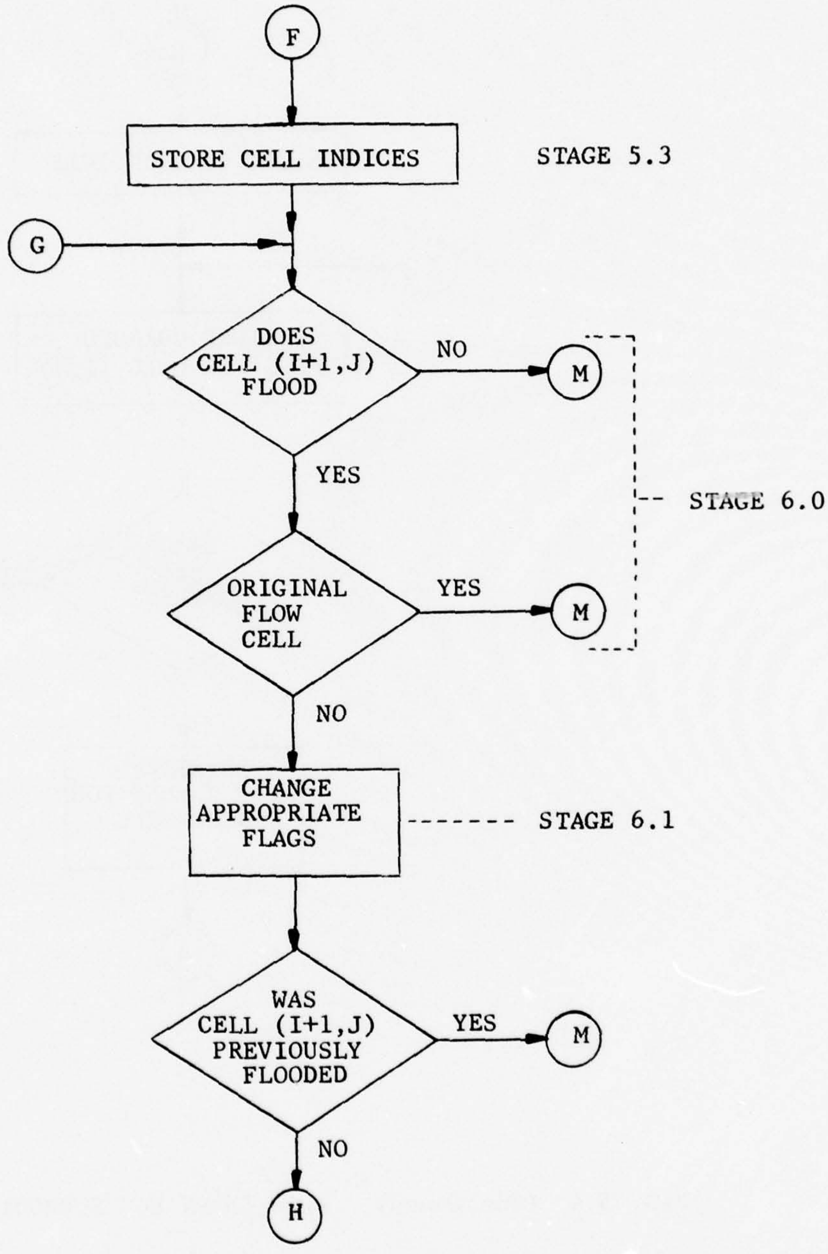


FIG. 5.4 (Continued) FLOW CHART FOR SUBROUTINE TRACE

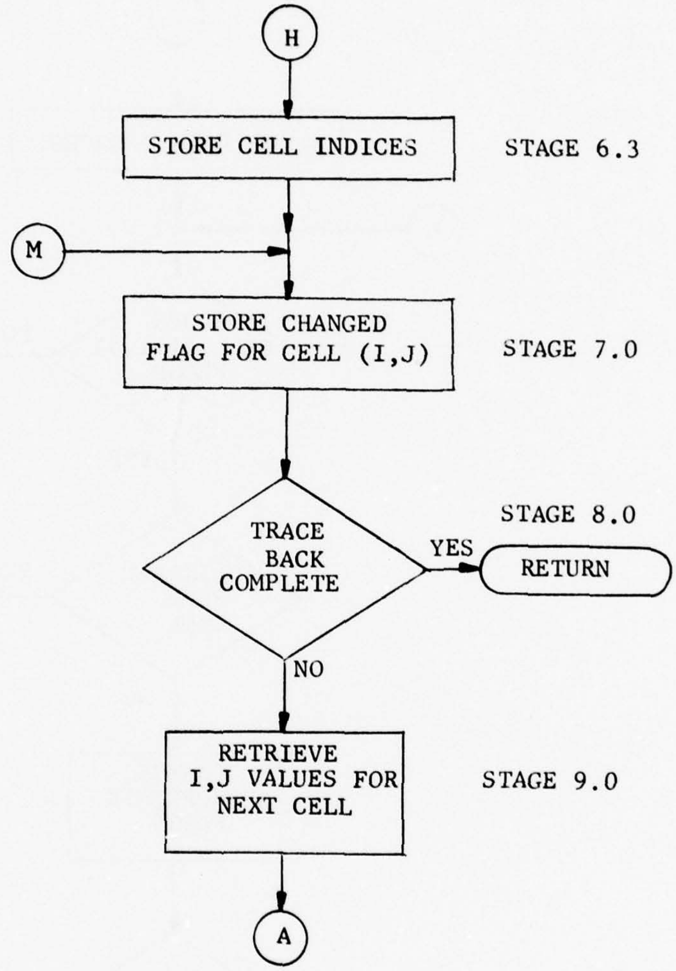


FIG. 5.4 (Continued) FLOW CHART FOR SUBROUTINE TRACE

TABLE 5.5

LISTING OF SUBROUTINE WIND

```
      SUBROUTINE WIND (TIME,XW,YW)                                      WND   1
C        ***************************************************************  WND   2
C        * THIS SUBROUTINE INPUTS THE WIND RECORD AND CALCULATES THE *  WND   3
C        * WIND EFFECT AS A FUNCTION OF TIME.  THE INPUT IS VIA DATA *  WND   4
C        * CARDS                                                     *  WND   5
C        * CONDITIONS.                                               *  WND   6
C        *                                                           *  WND   7
C        * N = NUMBER OF INPUT SETS                                  *  WND   8
C        *    =0 STANDARD CAL CONDITIONS                             *  WND   9
C        * ROT - ROTATION ANGLE TO MAKE WDIR W.R.T. POSITIVE X AXIS  *  WND  10
C        * WSPEED - WIND SPEED (MPH)                                 *  WND  11
C        * WDIR - WIND DIRECTION (DEGREES) W.R.T. POSITIVE X AXIS    *  WND  12
C        * WTIME - TIMES AT WHICH WIND CHANGES (MINUTES)             *  WND  13
C        ***************************************************************  WND  14
      DIMENSION WSPEED(10), WDIR(10), WTIME(10)                         WND  15
      DATA IT1/1/                                                       WND  16
      DATA IC/1/WC/33.5/                                                WND  17
      IF (IT1.EQ.0) GO TO 4                                             WND  18
C
C     *******************************************************************
C     STAGE 1 -- SET VARIABLES FOR CALM CONDITIONS FOR DEFALT
C     *******************************************************************
C
      WSPEED(1)=0.                                                      WND  19
      WSPEED(2)=0.                                                      WND  20
      WDIR(1)=0.                                                        WND  21
      WDIR(2)=0                                                         WND  22
      WTIME(1)=0.                                                       WND  23
      WTIME(2)=1.0E6
      IT1=0                                                             WND  25
C
C     *******************************************************************
C       STAGE 2 --  READ DATA FROM CARDS
C              WIND DIRECTIONS W.R.T. NORTH IN CLOCKWISE DIRECTION
C              AND IS DIRECTION FROM WHICH IT IS COMMING - THIS IS
C              HOW THE WIND IS REPORTED BY THE WEATHER BUREAU
C     *******************************************************************
C
      READ 1, N,ROT                                                     WND  26
```

TABLE 5.5 - continued

LISTING OF SUBROUTINE WIND

```
      IF (N.EQ.0) RETURN                                               WND  27
      READ 2, (WSPEED(I),WDIR(I),WTIME(I),I=1,N)                       WND  28
 1    FORMAT (I2,F10.1)                                                WND  29
 2    FORMAT (3F10.2)                                                  WND  30
C
C    **********************************************************************
C    STAGE 3 -- ROTATE WIND DIRECTION TO BE W.R.T. TO POSITIVE X-AXES AND
C             DIRECTION WIND IS BLOWING TOWARD
C    **********************************************************************
C
      DO 3 I=1,N                                                       WND  31
      WDIR(I)=360.0-WDIR(I)+ROT                                        WND  32A
    3 IF (WDIR(I) .GT. 360. ) WDIR(I)=WDIR(I)-360.                     WND  32B
      WTIME(N+1)=1.0E6
      IF ( TIME .GE. WTIME(1)) GO TO 9
C
C    **********************************************************************
C    STAGE 4 -- CHECK FOR ERROR IN INITIAL TIME AND PRINT ERROR MESSAGE
C    **********************************************************************
C
      PRINT 8,TIME,WTIME(1)
    8 FORMAT(1H1,20(/),5X,'*****************************'/5X,'* ERROR IN
     1WIND'/5X,'* TIME .LT. WTIME(1)'/5X,'*',F9.2,' .LT. ',F9.2/5X,
     2'* PROGRAM STOPED'/5X,'*****************************')
      STOP
C
C    **********************************************************************
C    STAGE 5 -- FIND INDICATOR VALUE FOR INITIAL TIME OF DAY
C    **********************************************************************
C
    9 DO 10 I=2,N
      IC=I-1
      IF ( TIME .GE. WTIME(I-1) .AND. TIME .LT. WTIME(I) ) GO TO 13
   10 CONTINUE
C
C    **********************************************************************
C    STAGE 6 -- PRINT ERROR MESSAGE -- INITIAL TIME TO LARGE
C    **********************************************************************
C
      PRINT 11,TIME
   11 FORMAT(1H1,20(/),5X,'*****************************'/5X,
     1'* TIME TO LARGE'/5X, '* TIME =',F9.2/5X,
     2'* PROGRAM STOPED'/5X,'*****************************')
```

TABLE 5.5 - continued

LISTING OF SUBROUTINE WIND

```
      STOP
   13 RETURN
C
C    ************************************************************************
C     STAGE 7 - THE FOLLOWING SECTION CALCULATES THE WIND DRAG
C    ************************************************************************
C
4     IF (TIME.LT.WTIME(IC)) RETURN                                       WND  34
      IF (WSPEED(IC).GE.WC) GO TO 5                                       WND  35
      CK=0.5*SQRT(1609.0*WSPEED(IC)/3600.0)*1.0E-3                        WND  36
      GO TO 6                                                             WND  37
5     CK=2.6E-3                                                           WND  38
6     XW=1.2E-3*CK*WSPEED(IC)*WSPEED(IC)*COS(WDIR(IC)*3.1416/180.)*7744.WND  39
      YW=1.2E-3*CK*WSPEED(IC)*WSPEED(IC)*SIN(WDIR(IC)*3.1416/180.)*7744.WND  40
      PTIME=TIME/60.0                                                     WND  41
      PRINT 7, PTIME,WSPEED(IC),WDIR(IC),XW,YW                            WND  42
7     FORMAT (10X,'THERE HAS BEEN A CHANGE IN WIND CONDITIONS AT TIME',FWND  43
     16.2,' HOURS.'/10X,'NEW PARAMETERS:  SPEED',F5.1,' MPH'/17X,'DIRECTWND  44
     2ION',F7.1,' DEGREES'/17X,'XW',F7.3,' SQ. FT./SQ. MIN.'/17X,'YW',F7WND  45
     3.3,' SQ. FT./SQ. MIN.'//)                                           WND  46
      IC=IC+1                                                             WND  47
      RETURN                                                              WND  49
      END                                                                 WND  50-
```

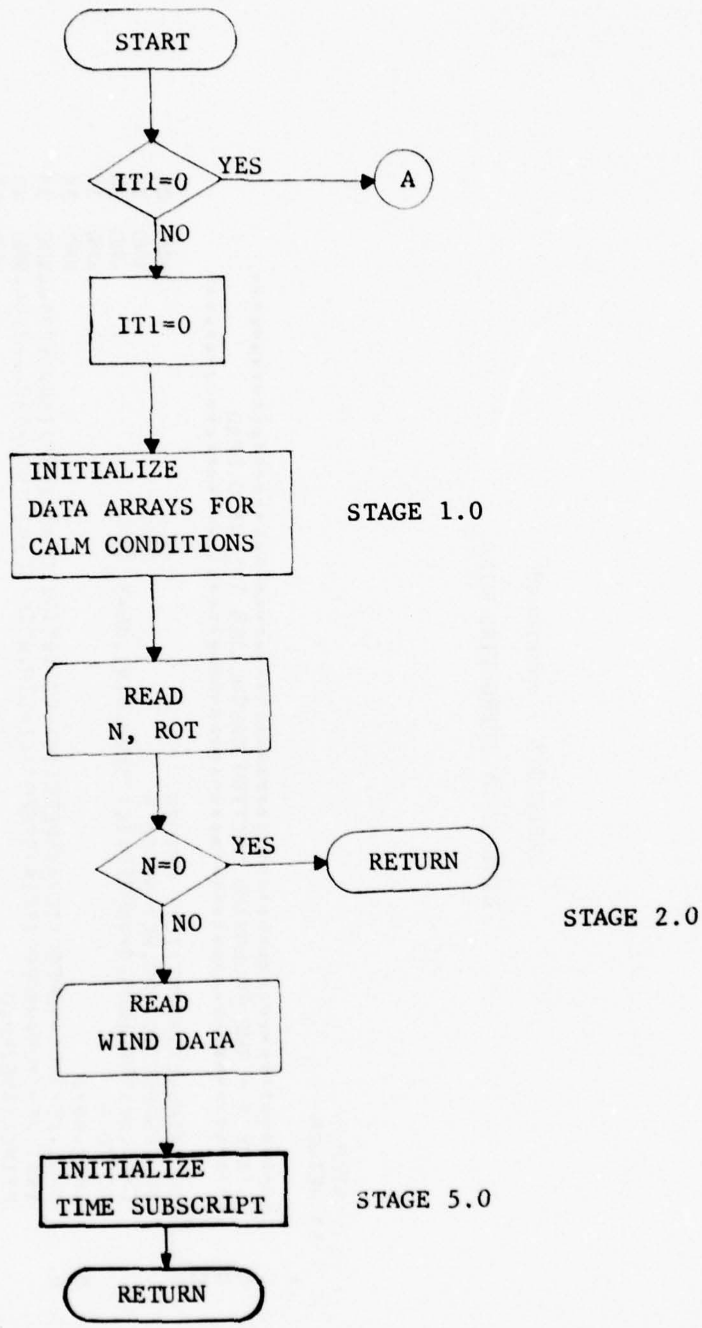


FIG. 5.5 FLOW CHART FOR SUBROUTINE WIND

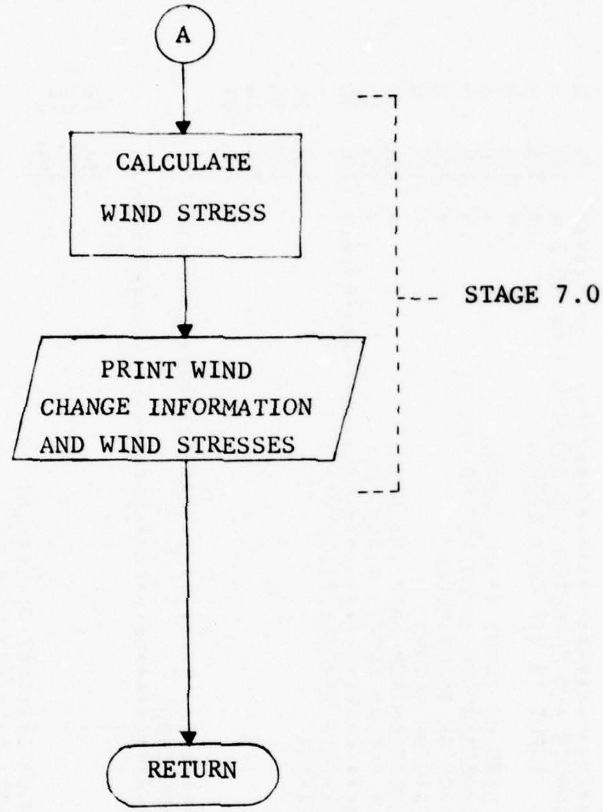


FIG. 5.5 (Continued) FLOW CHART FOR SUBROUTINE WIND

TABLE 5.6

LISTING OF SUBROUTINE TIDAL2

```
      SUBROUTINE TIDAL2(T,H)
C        ***********************************************************  TDL   2
C        * THIS SUBROUTINE INPUTS AND COMPUTES THE DRIVING TIDE AS A *  TDL   3
C        * FUNCTION OF TIME.  INPUT IS VIA DATA CARDS.               *  TDL   4
C        *                                                           *  TDL   5
C        * NDAY - NUMBER OF DAYS OF INPUT TIDAL RECORD               *  TDL   6
C        * DSHIFT - DATUM CORRECTION ON INPUT RECORD (FEET)          *  TDL   7
C        * LOC - IDENTIFICATION OF INPUT RECORD                      *  TDL   8
C        * DATE - CALENDAR DATE OF RECORD                            *  TDL   9
C        * TIMCOR - TIME CORRECTION FOR DATE (MINUTES)               *  TDL  10
C        * TD - ACTUAL HOURLY RECORD (FEET) W.R.T. MSL               *  TDL  11
C        ***********************************************************  TDL  12
      DIMENSION TR(480),TD(48,10), TIMCOR(10),DATE(10)
      EQUIVALENCE (TR(1),TD(1,1))                                       TDL  14
      DATA IMP/1/                                                       TDL  15
      IF (IMP.EQ.0) GO TO 11                                            TDL  16
      IMP=0                                                             TDL  17
C
C     ****************************************************************
C     STAGE 1 -- READ DATA FROM CARDS
C     ****************************************************************
C
      READ 1, NDAY,DSHIFT                                               TDL  18
1     FORMAT (I10,F10.0)                                                TDL  19
      DO 3 J=1,NDAY                                                     TDL  20
      READ 2,LOC,DATE(J),TIMCOR(J),(TD(I,J),I=1,48)
    2 FORMAT(A4,A6,F6.0,4X,12F4.2,3(/20X,12F4.2))
C
C     ****************************************************************
C     STAGE 2 -- SUBTRACT DATUM CORRECTION
C     ****************************************************************
C
      DO 3 I=1,48
3     TD(I,J)=TD(I,J)-DSHIFT                                            TDL  24
C
C     ****************************************************************
C     STAGE 3 -- ECHO PRINT DATA
C     ****************************************************************
C
```

TABLE 5.6 - continued

LISTING OF SUBROUTINE TIDAL2

```
      PRINT 4, LOC,DSHIFT                                               TDL  25
4     FORMAT (//10X,'INPUT TIDE DATA(IN FEET FROM MSL) ',10X,'GAGE AT ',TDL  26
     1A4/10X,'NOTE: A DATUM SHIFT OF ',F5.2,' HAS BEEN APPLIED'/)      TDL  27
      PRINT 5, (DATE(I),I=1,NDAY)                                       TDL  28
5     FORMAT (5X,18(1XA6))                                              TDL  29
      PRINT 6                                                           TDL  30
6     FORMAT (' HOUR')                                                  TDL  31
      DO 7 I=0,47
      RI=I*.5
    7 PRINT 8,RI,(TD(I+1,J),J=1,NDAY)
    8 FORMAT(5X,F4.1,18F7.2)
      PRINT 9, (TIMCOR(I),I=1,NDAY)                                     TDL  35
9     FORMAT (/5X,18F7.1)                                               TDL  36
      PRINT 10                                                          TDL  37
10    FORMAT (10X,'TIME CORRECTION(MIN) TO BE APPLIED'//)               TDL  38
C
C    ***********************************************************************
C    STAGE 5 -- FIND TIDE LEVEL BY LINEAR INTERPOLATION
C    ***********************************************************************
C
11    THOUR=T/60.0                                                      TDL  39
      IDAY=THOUR/24+1                                                   TDL  40
      THOUR=THOUR-TIMCOR(IDAY)/60.0                                     TDL  41
      IT=THOUR*2.0
      FRAC=THOUR+THOUR-IT
      H=TR(IT+1)+FRAC*(TR(IT+2)-TR(IT+1))                               TDL  44
      RETURN                                                            TDL  45
      END                                                               TDL  46-
```

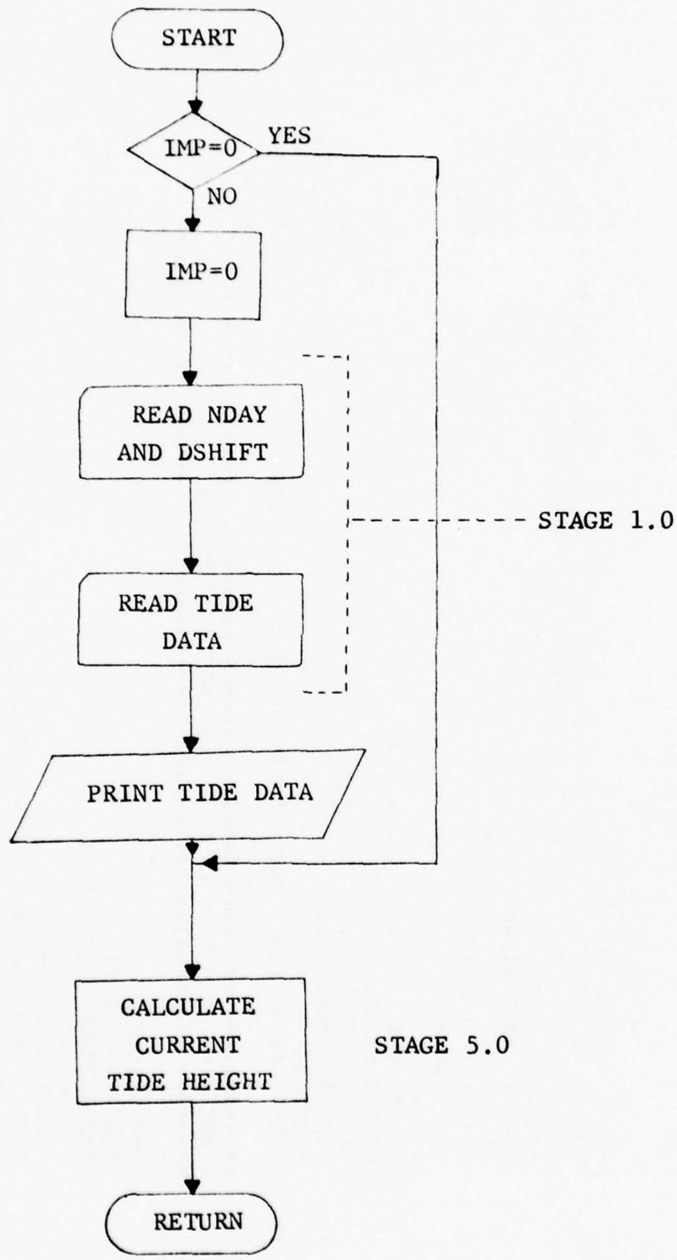


FIG. 5.6 FLOW CHART FOR SUBROUTINE TIDAL2

TABLE 5.7

CELLS USED TO FIND VELOCITIES AND TIDES FOR STATIONS USED

Station Number	North (West)	Center	South (East)	Tide
1	(16,7)	$\frac{(15,7)+(16,7)}{2}$	$\frac{(15,7)+(15,8)}{2}$	$\frac{(16,6)+(16,7)}{2}$
2	$\frac{(16,14)+(16,15)}{2}$	$\frac{(15,14)+(15,15)}{2}$	$\frac{(15,14)+(14,15)}{2}$	(18,7)
3	$\frac{(16,22)+(16,23)}{2}$	$\frac{(15,22)+(16,23)}{2}$	(15,23)	(15,23)
4	$\frac{(29,18)+(28,18)}{2}$	(28,17)	$\frac{(28,16)+(28,17)}{2}$	(28,16)
5	$\frac{(3,19)+(4,19)}{2}$	$\frac{(3,18)+(4,19)}{2}$	$\frac{(3,18)+(4,18)}{2}$	(3,18)

TABLE 5.8

VALUES OF IOPP AND CORRESPONDING
VARIABLE FOR USE IN CALL TO MODPRI

IOPP	Variable	Units
1	Flag field	---
2	Water depths	feet
4	Velocities in x-direction	ft/second
8	Velocities in y-direction	ft/second
16	Water levels	feet (wrt datum)
32	Flows in x-direction	ft^2/minute
64	Flows in y-direction	ft^2/minute

TABLE 5.9

LISTING OF SUBROUTINE PRINT1

```
      SUBROUTINE PRINT1(TIM,TIDE)
C     ************************************************************  PRI   2
C     * THIS SUBROUTINE IS THE MAIN ROUTINE FOR OUTPUT TO THE        *
C     * PRINTER.  THE OUTPUT OPERATIONS ARE CONTROLLED THROUGH       *
C     * TWO ENTRY POINTS.  A THIRD ENTRY POINT IS FOR                *
C     * INITIALIZATION.                                              *
C     *                                                              *
C     * ENTRY POINT PRINT1 - THIS ENTRY POINT HAS THREE FUNCTIONS    *
C     *    1. PRINTUNG OF TIDAL ELEVATION AND MASNITUDE AND          *
C     *    INDICES ARE READ FROM CARDS VIA ENTRY POINT PIPREP.       *
C     *    2. PRINTING A TABLE OF TIDAL ELEVATIONS AND VELOCITIES    *
C     *       AT THE 5 STATIONS AT MASONBORO INLET                   *
C     *    3. STOREING OF TIDAL ELEVATIONS AND VELOCITIES AT THE     *
C     *       5 STATIONS PRINTED UNDER 2 FOR PUNCHING WHEN TLIM      *
C     *       IS REACHED.                                            *
C     *    WHICH OF THE THREE DONE IS CONTROLED BY THE PARAMETER     *
C     *    IPCH.  THERE ARE FOUR (4) POSSIBILITIES                   *
C     *       IPCH = 0 - INFORMATION FOR SPECIFIED CELLS  AND        *
C     *                  STATIONS PRINTED BUT NOTHING STORED,        *
C     *                  I.E.   1 AND 2 ABOVE DONE                   *
C     *            = 1 - ALL OF THE ABOVE DONE                       *
C     *            = 2 - SPECIFIED CELLS NOT PRINTED BUT STATION     *
C     *                  INFORMATION IS PRINTED AND STORED,I.E.      *
C     *                  2 AND 3 ABOVE DONE                          *
C     *            = 3 - SPECIFIED CELLS PRINTED AND STATION         *
C     *                  INFORMATION STORED FOR PUNCHING,I.E.        *
C     *                  1 AND 3 ABOVE DONE.                         *
C     *                                                              *
C     * ENTRY POINT PIPREP - THIS ENTRY POINT READS IN THE NUMBER    *
C     *    OF SEPCIFIED CELLS FOR WHICH INFORMATION IS TO BE         *
C     *    PRINTED (MAXIMUM OF 17)AND THEIR INDICES .                *
C     *                                                              *
C     * ENTRY POINT MODPRI - THROUGH THIS ENTRY POINT ENTIRE         *
C     *    ARRAYS OF 7 VARIABLES MAY BE PRINTED.  THE VARIABLES      *
C     *    TO BE PRINTED ANE CONTROLLED BY THE PARAMETER IOP.        *
C     *    TO FIND THE VALUE OF IOP ADD THE NUMBERS CORRESPONDING    *
C     *    TO THE VARIABLES TO BE PRINTED TOGETHER.  THE             *
C     *    CORRESPONDENCE IS                                         *
C     *          1 - FLAG FIELD                                      *
```

TABLE 5.9 - continued

LISTING OF SUBROUTINE PRINT1

```
C        *           2 - DEPTH FIELD
C        *           4 - X COMPONENT OF VELOCITY                        *
C        *           8 - Y COMPONENT OF VELOCITY                        *
C        *          16 - WATER LEVEL (H)                                *
C        *          32 - X COMPONENT OF FLOW                            *
C        *          64 - Y COMPONENT OF FLOW                            *
C        *                                                              *
C        *                                                              *  PRI   7
C        * IPL - NUMBER OF CELLS FOR WHICH INFORMATION IS TO BE         *  PRI   8
C        *        DISPLAYED (MUST NOT EXCEED 17)                        *  PRI   9
C        * TITLE - ARRAY OF SEVEN (7) CHARACTER LABLES FOR CELLS        *  PRI  10
C        * IP - ARRAY OF HORIZONTAL CELL NUMBERS                        *  PRI  11
C        * JP - ARRAY OF VERTICAL CELL NUMBERS                          *  PRI  12
C        ****************************************************************  PRI  13
      COMMON /A/QX(30,25),QY(30,25),D(30,25),IFLAG(30,25),H(30,25),
     1IMAX,JMAX,DELTA,JB,JBR,IBU,PUNV(5,4,50),NPU,PCHT(50),IPCH
      DATA NPU/0/
      DIMENSION SV(5,3),TS(5),ID(5,3,2),JD(5,3,2)
      DATA ID/16,16,16,29,3,15,15,15,28,3,15,15,15,28,3,16,16,16,28,
     14,16,15,16,28,4,15,14,15,28,4/
      DATA JD/7,14,22,18,19,7,14,22,17,18,7,14,23,16,18,7,15,23,18,19,7,15,23,17
     115,23,17,19,8,15,23,17,18/
      DIMENSION VEL(17), DIR(17), HP(17), IP(17), JP(17), TITLE(2,17)  PRI  15
      IF ( IPCH .EQ. 2 ) GO TO 10
C
C     ******************************************************************
C      STAGE 1.0 - PRINT TIDAL ELEVATIONS AND VELOCITIES AT
C                  SPECIFIED CELLS
C     ******************************************************************
C
      IF( NPRINT .NE. 10 ) GO TO 1
      NPRINT=0                                                         PRI  18
      PRINT 8, ((TITLE(J,I),J=1,2),I=1,IPL)                            PRI  19
 1    NPRINT=NPRINT+1                                                  PRI  20
      TIMP=TIM/60.
C
C     ******************************************************************
C      STAGE 1.1 - CALCULATE VELOCITIES AT SPECIFIED CELLS AND STORE
C     ******************************************************************
C
      DO 2 K=1,IPL                                                     PRI  22
      K1=IP(K)                                                         PRI  23
      K2=JP(K)
```

TABLE 5.9 - continued

LISTING OF SUBROUTINE PRINT1

```
      HP(K)=H(K1,K2)                                                    PRI  25
      UU=0.5*(QX(K1,K2)+QX(K1-1,K2))/(60.0*D(K1,K2))                    PRI  26
      VV=0.5*(QY(K1,K2)+QY(K1,K2-1))/(60.0*D(K1,K2))                    PRI  27
      VEL(K)=SQRT(UU*UU+VV*VV)                                          PRI  28
      IF (VEL(K).EQ.0.0) GO TO 2                                        PRI  29
      DIR(K)=ATAN2(VV,UU)*180.0/3.14159                                 PRI  30
2     CONTINUE                                                          PRI  31
C
C     ******************************************************************
C     STAGE 1.2 - PRINT VELOCITIES AND TIDAL ELEVATIONS AT SPECIFIED CELLS
C     ******************************************************************
C
      PRINT 3, TIMP,TIDE,(HP(K),K=1,IPL)                                PRI  32
3     FORMAT (/1X,F5.1,18F7.3)                                          PRI  33
      PRINT 4, (VEL(K),K=1,IPL)                                         PRI  34
4     FORMAT (13X,17F7.3)                                               PRI  35
      PRINT 5, (DIR(K),K=1,IPL)                                         PRI  36
5     FORMAT (13X,17F7.1)                                               PRI  37
C
C    *******************************************************************
C     STAGE 2.0 - CALCULATE VELOCITIES AND TIDAL ELEVATIONS AT STATIONS
C     ******************************************************************
C
   10 DO 50 I=1,5
      DO 40 J=1,3
      I1=ID(I,J,1)
      J1=JD(I,J,1)
      I2=ID(I,J,2)
      J2=JD(I,J,2)
      U1=0.5*(QX(I1,J1)+QX(I1-1,J1))/(60.0*D(I1,J1))
      U2=0.5*(QX(I2,J2)+QX(I2-1,J2))/(60.0*D(I2,J2))
      V1=0.5*(QY(I1,J1)+QY(I1,J1-1))/(60.0*D(I1,J1))
      V2=0.5*(QY(I2,J2)+QY(I2,J2-1))/(60.0*D(I2,J2))
      VEL1=SQRT(U1*U1+V1*V1)
      VEL2=SQRT(U2*U2+V2*V2)
      SV(I,J)=(VEL1+VEL2)*.5
      IF ( I .GT. 3 ) GO TO 35
      IF ( V1 .LT. 0. ) SV(I,J)=-SV(I,J)
      GO TO 40
   35 IF ( I .EQ. 5 ) GO TO 36
      IF ( U1 .LT. 0.) SV(I,J)=-SV(I,J)
      GO TO 40
   36 IF ( U1 .GT. 0. ) SV(I,J)=-SV(I,J)
```

TABLE 5.9 - continued

LISTING OF SUBROUTINE PRINT1

```
   40 CONTINUE
   50 CONTINUE
      TS(1)=(H(16,6)+H(16,7))*.5
      TS(2)=H(18,17)
      TS(3)=H(15,23)
      TS(4)=H(28,16)
      TS(5)=H(4,18)
      IF ( IPCH .EQ. 3 ) GO TO 65
C
C     ****************************************************************
C     STAGE 2.1 - PRINT VELOCITIES AND TIDAL ELEVATIONS AT STATIONS
C     ****************************************************************
C
      PRINT 61,(I,TS(I),(SV(I,J),J=1,3),I=1,3)
   61 FORMAT(////34X,'VELOCITIES (FEET/SEC)'/
     1,11X,'STATION   TIDE(FT)  NORTH    CENTER      SOUTH'/
     23(I15,F12.2,3F10.2/))
      PRINT 63,(I,TS(I),(SV(I,J),J=1,3),I=4,5)
   63 FORMAT( 32X,'WEST     CENTER      EAST'/3(I15,F12.2,3F10.2/))
      IF ( IPCH .EQ. 0 ) RETURN
  65  NPU=NPU+1
C
C     ****************************************************************
C     STAGE 2.2 - STORE VELOCITIES AND TIDAL ELEVATIONS AT STATIONS
C                 FOR PUNCHING LATER
C     ****************************************************************
C
      PCHT(NPU)=TIMP
      DO 64 I=1,5
      PUNV(I,4,NPU)=TS(I)
      DO 64 J=1,3
      PUNV(I,J,NPU)=SV(I,J)
   64 CONTINUE
      RETURN                                                          PRI  38
C
C    *****************************************************************
C     STAGE 3.0 - ENTRY P1PREP
C                 READ IN NUMBER OF SPECIFIED CELLS AND THEIR INDICES
C     ****************************************************************
C
      ENTRY P1PREP                                                    PRI  39
      NPRINT=0                                                        PRI  40
      READ 7, IPL                                                     PRI  41
```

TABLE 5.9 - continued

LISTING OF SUBROUTINE PRINT1

```
      READ 6, ((TITLE(J,I),J=1,2),I=1,IPL)
6     FORMAT (10(A6,A1))
      READ 7, (IP(I),JP(I),I=1,IPL)
7     FORMAT (16I5)
      PRINT 8, ((TITLE(J,I),J=1,2),I=1,IPL)
8     FORMAT (1H1,1X,'TIME   TIDE',17(A6,A1))
      RETURN
C
C    *****************************************************************
C     STAGE 4.0 - ENTRY MODPRI
C               PRINT INDICATED ARRAYS -- SEE ABOVE FOR VALUE OF IOP
C    *****************************************************************
C
      ENTRY MODPRI(IOP,TTT)
      COMMON /B/DUM(30,25),DX,DY
      TIME=TTT/60.
      J2=JMAX/2.
      J21=J2+1
      DY60=DY/60.
      DX60=DX/60.
      IF (JMAX .LE. 15 ) J2=JMAX
      IF ( FLD(35,1,IOP) .NE. 1 ) GO TO 198
C
C    *****************************************************************
C     STAGE 4.1 - PRINT FLAG FIELD
C    *****************************************************************
C
      PRINT 99,TIME
   99 FORMAT(1H1,//20X,'FLAG FIELD    TIME=',F8.3,' HR.'///)
      PRINT 249,(J,J=1,JMAX)
      DO 100 I=1,IMAX
      PRINT 101,(I,(IFLAG(I,J),J=1,JMAX))
  101 FORMAT(3X,I2,2X,25(I4,'*'))
  100 CONTINUE
  198 IF ( FLD(34,1,IOP) .NE. 1 ) GO TO 255
C
C    *****************************************************************
C     STAGE 4.2 - PRINT DEPTH FIELD
C    *****************************************************************
C
      PRINT 199,TIME
      PRINT 249,(J,J=1,JMAX)
      DO 250 I=1,IMAX
```

TABLE 5.9 - continued

LISTING OF SUBROUTINE PRINT1

```
      PRINT 201 ,(I,(D(I ,J),J=1,JMAX))
  201 FORMAT(3X,I2,2X,25F5.1)
  250 CONTINUE
  249 FORMAT(5X,25I5/)
  199 FORMAT(1H1,20X,'DEPTH FIELD -- ALL DRY LAND PRINTED AS *****'/
     120X,'TIME=',F8.3,' HR.'///)
  255 IF ( FLD(33,1,IOP) .NE. 1 ) GO TO 400
C
C     ******************************************************************
C     STAGE 4.3 - PRINT X - VELOCITY
C     ******************************************************************
C
      PRINT 302,TIME
  302 FORMAT(1H1,//'   ******* X-VELOCITY (FT/SEC) *******    TIME=',
     1F8.3,' HR.'///)
      DO 800 I=1,IMAX
      DO 800 J=1,JMAX
      DUM(I,J)=QX(I,J)/(D(I,J)*60.)
  800 CONTINUE
      PRINT 98,(J,J=1,J2)
   98 FORMAT(15I8)
      DO 300 I=1,IMAX
      PRINT 301,I,(DUM(I,J),J=1,J2)
  300 CONTINUE
      IF (JMAX .LE. 15) GO TO 400
      PRINT 97
      PRINT 98,(J,J=J21,JMAX)
   97 FORMAT(1H1//'   ******* X-VELOCITY CONTINUED *******'///)
      DO 3000 I=1,IMAX
      PRINT 301,I,(DUM(I,J),J=J21,JMAX)
 3000 CONTINUE
  301 FORMAT(I4,15E8.3)
  400 IF ( FLD(32,1,IOP) .NE. 1 ) GO TO 410
C
C     ******************************************************************
C     STAGE 4.4 - PRINT Y - VELOCITY
C     ******************************************************************
C
      PRINT 303,TIME
  303 FORMAT(1H1//'   ******* Y-VELOCITY (FT/SEC) *******   TIME=',
     1F8.3,' HR.'///)
      DO 810 I=1,IMAX
      DO 810 J=1,JMAX
```

TABLE 5.9 - continued

LISTING OF SUBROUTINE PRINT 1

```
      DUM(I,J)=QY(I,J)/(D(I,J)*60.)
  810 CONTINUE
      PRINT 98,(J,J=1,J2)
      DO 304 I=1,IMAX
      PRINT 301,I,(DUM(I,J),J=1,J2)
  304 CONTINUE
      IF (JMAX .LE. 15) GO TO 410
      PRINT 96
   96 FORMAT(1H1//'    ******* Y-VELOCITY CONTINUED *******'///)
      PRINT 98,(J,J=J21,JMAX)
      DO 3004 I=1,IMAX
      PRINT 301,I,(DUM(I,J),J=J21,JMAX)
 3004 CONTINUE
  410 IF (FLD(31,1,IOP) .NE. 1) GO TO 420
C
C    *****************************************************************
C     STAGE 4.5 - PRINT WATER LEVEL
C    *****************************************************************
C
      PRINT 305,TIME,(J,J=1,JMAX)
  305 FORMAT(1H1//'    ******* H-WATER LEVEL (FT) *******   TIME=',
     1F8.3,' HR.'//5X,25I5)
      DO 306 I=1,IMAX
      PRINT 307,I,(H(I,J),J=1,JMAX)
  306 CONTINUE
  307 FORMAT(I5,25F5.3)
  420 IF ( FLD(30,1,IOP) .NE. 1 ) GO TO 430
C
C    *****************************************************************
C     STAGE 4.6 - PRINT X FLOWS
C    *****************************************************************
C
      PRINT 401,TIME,(J,J=1,J2)
  401 FORMAT(1H1//'    ******* X-FLOW (CU FT/SEC) *******    TIME=',
     1F8.3,' HR.'///15I8)
      DO 402 I=1,IMAX
      DO 402 J=1,JMAX
  402 DUM(I,J)=QX(I,J)*DX60
      DO 405 I=1,IMAX
  405 PRINT 301,I,(DUM(I,J),J=1,J2)
      IF (JMAX .LE. 15) GO TO 430
      PRINT 403,(J,J=J21,JMAX)
  403 FORMAT(1H1//'    ******* Y-FLOW CONTINUED *******'///15I8)
```

TABLE 5.9 - continued

LISTING OF SUBROUTINE PRINT1

```
      DO 404 I=1,IMAX
  404 PRINT 301,I,(DUM(I,J),J=J21,JMAX)
  430 IF (FLD(29,1,IOP) .NE. 1)RETURN
C
C     ******************************************************************
C      STAGE 4.8 - PRINT Y FLOWS
C     ******************************************************************
C
      PRINT 411,TIME,(J,J=1,J2)
  411 FORMAT(1H1//'    ******* Y-FLOW (CU FT/SEC) *******  TIME=',F8.3,
     1' HR.'///15I8)
      DO 412 I=1,IMAX
      DO 412 J=1,JMAX
  412 DUM(I,J)=QY(I,J)*DY60
      DO 415 I=1,IMAX
  415 PRINT 301,I,(DUM(I,J),J=1,J2)
      IF (JMAX .LE. 15) RETURN
      PRINT 413,(J,J=J21,JMAX)
  413 FORMAT(1H1//'    ******* Y-FLOW CONTINUED *******   '///)
      DO 414 I=1,IMAX
  414 PRINT 301,I,(DUM(I,J),J=J21,JMAX)
      RETURN
      END
```

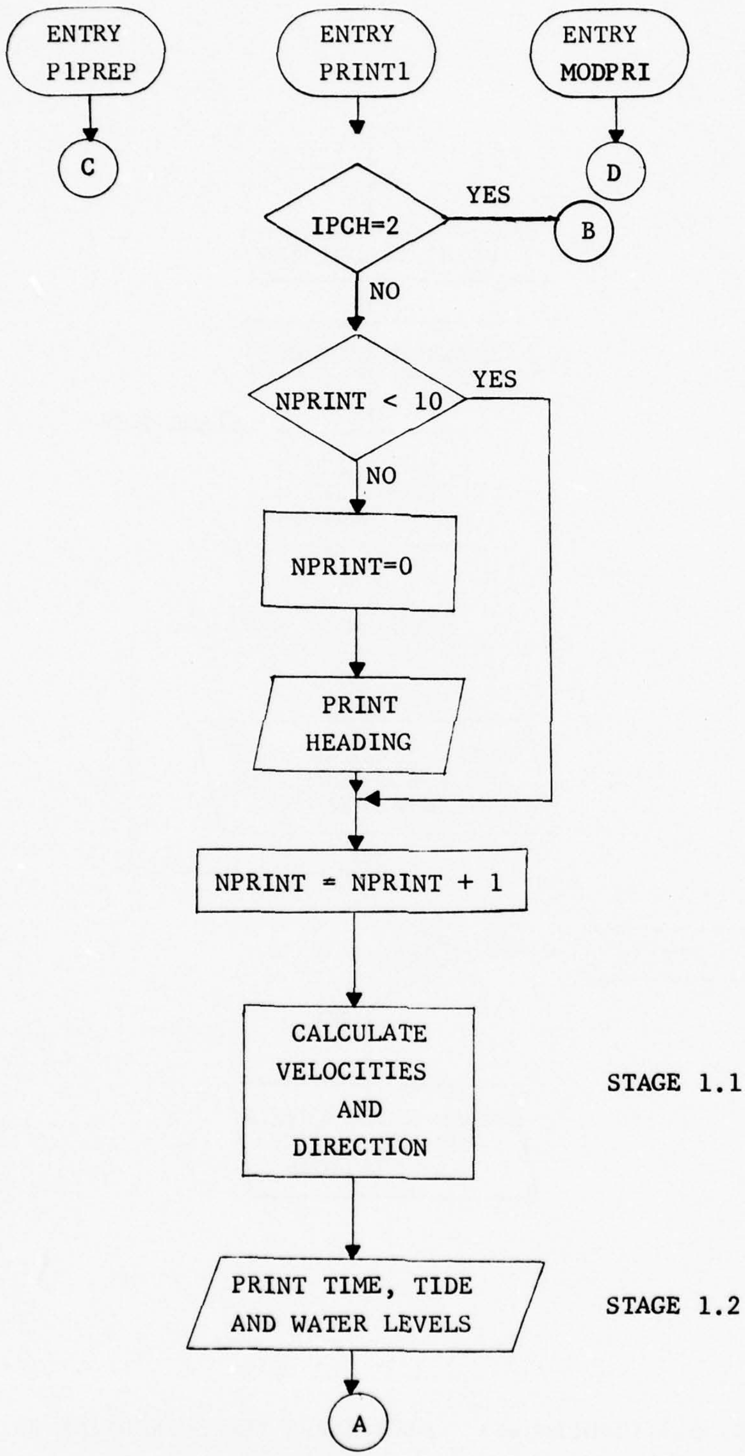


FIG. 5.7 FLOW CHART FOR SUBROUTINE PRINT1

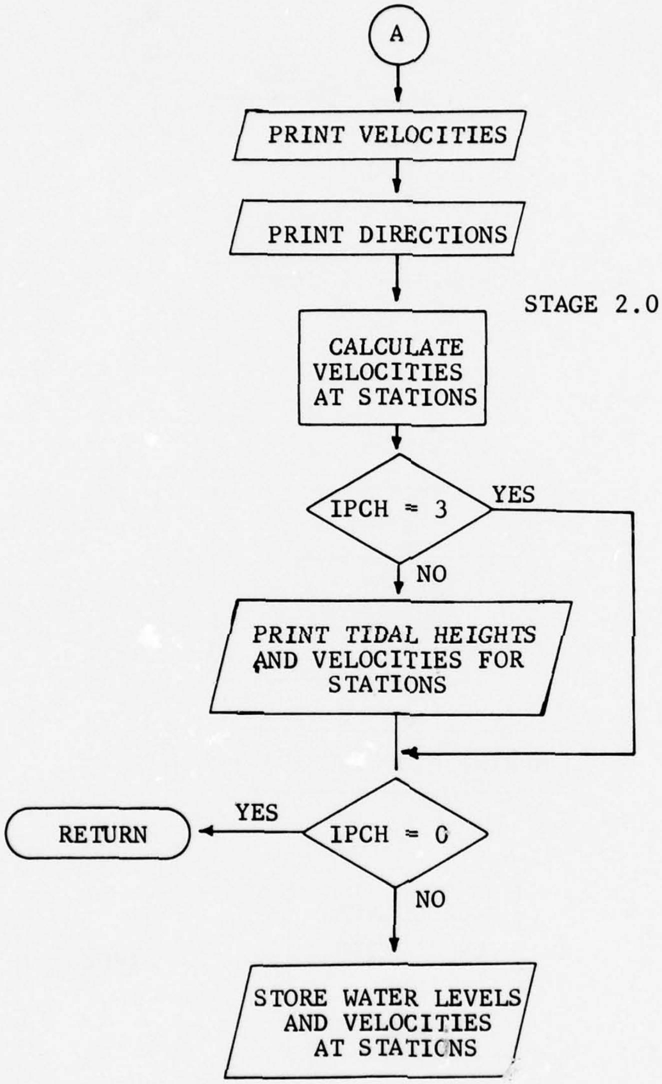


FIG. 5.7 (Continued) FLOW CHART FOR SUBROUTINE PRINT1

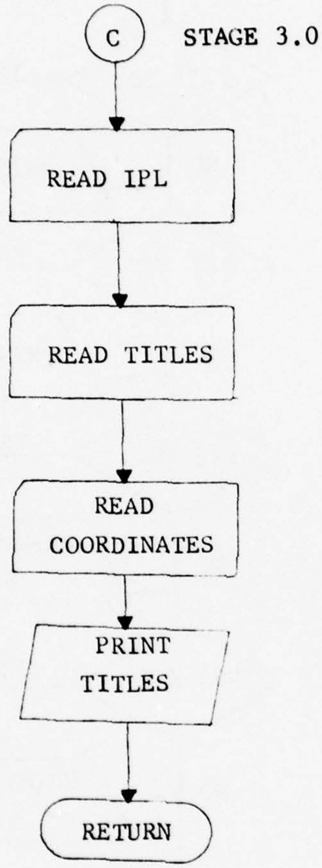
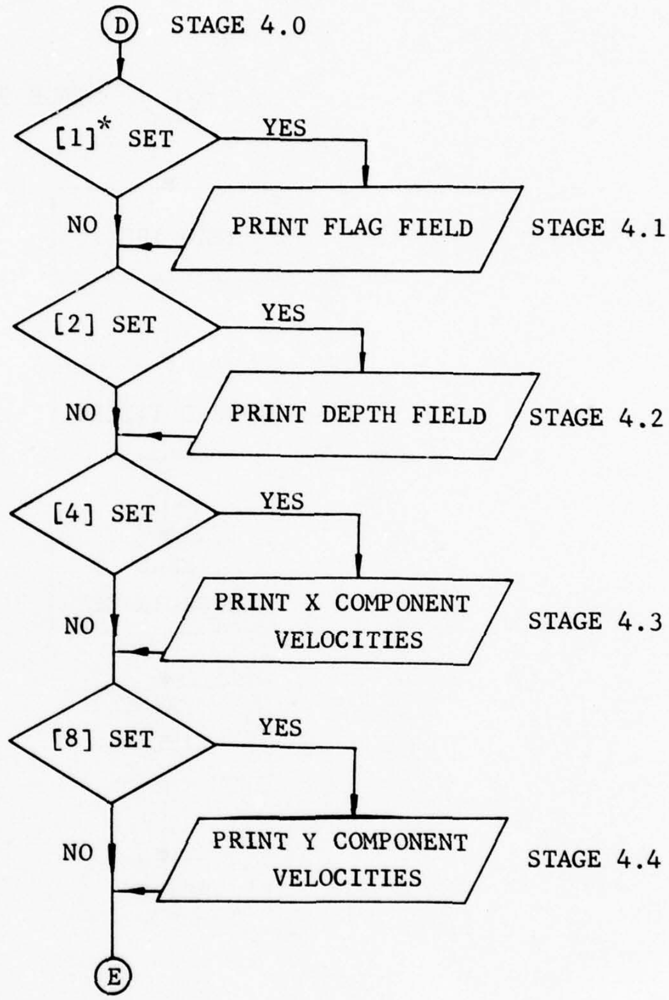


FIG. 5.7 (Continued) FLOW CHART FOR SUBROUTINE PRINT1



*[1] = bit position corresponding to the value contained in brackets

FIG. 5.7 (Continued) FLOW CHART FOR SUBROUTINE PRINT1

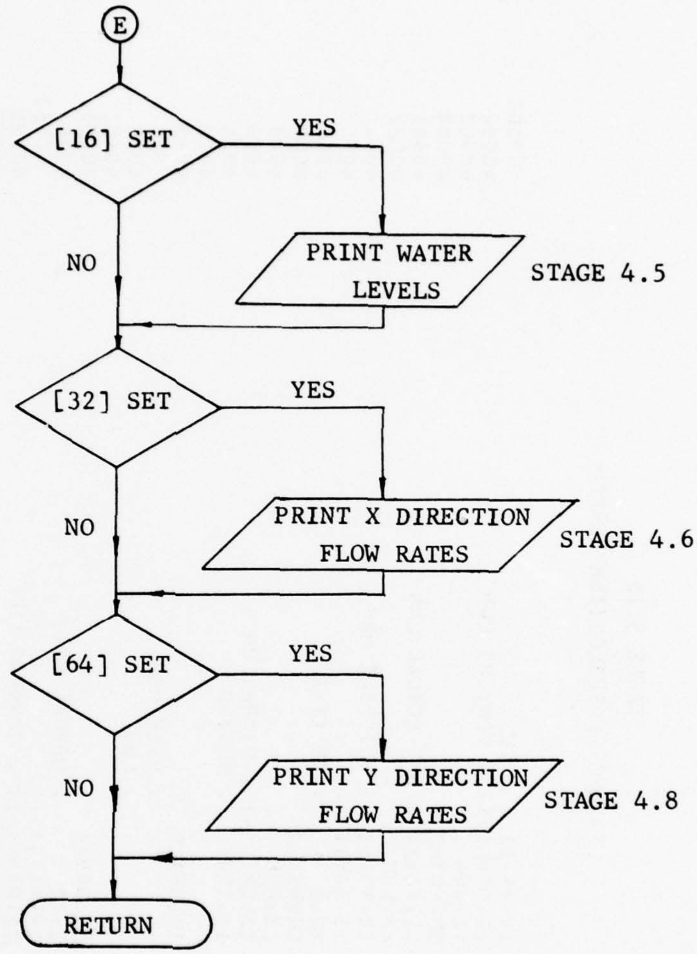


FIG. 5.7 (Continued) FLOW CHART FOR SUBROUTINE PRINT1

TABLE 5.10

LISTING OF SUBROUTINE SEARCH

```
/
SRCHOF*   LA        12,*1,11    . LUN                          40001
          LA        12,NTAB$,12 . CONVERT LUN                  40002
          SA        12,LUN .                                   40003
          J         SRCHOT+2 .                                 40004
SRCHOT*   LA        12,*1,11 . GET ACTUAL LUN                  40021
          SA        12,LUN      .                              40023
          LA        12,*0,11    . GET FILE NBR                 4004
          SX        11,SAV11    .                              4005
          TE,016    12,1        . IS IT 1                      4006
          J         SRCHIN+2                                   4007
          LA        13,LUN .                                   4008
          SA,1      13,SRCH1+1 . STORE LUN                     4009
SRCH1     LMJ       11,TRW$     . REWIND                       4011
          +0 . UNIT                                            4012
          LMJ       11,WFILE    .                              4013
SRCH2      (1)                  . FILE NBR 1                   4014
          +LUN                  . LUN                          4015
          NOP .                                                40151
          LX        11,SAV11                                   40151
          J         3,11        . NORMAL EXIT                  4016
SRCHIF*   LA        12,*1,11    .                              40161
          LA        12,NTAB$,12 . CONVERT LUN                  40162
          SA        12,LUN .                                   40163
          J         SRCHIN+2                                   401631
SRCHIN*   LA        12,*1,11.                                  40164
          SA        12,LUN                                     40166
          SX        11,SAV11 .                                 40167
          SZ        67          . CLEAR REWOUND FLAG           4018
          SX        11,SAV11    . SAVE X11                     4019
          LA        13,LUN .                                   4020
          SA,1      13,SRCH3    . STORE LUN                    4021
          SA,1      13,SRCH4+1 .                               4022
          SA,1      13,SRCH5+1 .                               4023
          SA,1      13,SRCH6    .                              4024
          SA,1      13,SRCH7+1                                 4025
          SA,1      13,SRCH8    .                              4026
          SA,1      13,SRCH8A .                                40261
          SA,1      13,SRCH8B                                  40262
```

TABLE 5.10 - continued

LISTING OF SUBROUTINE SEARCH

```
         SA,1     13,SRCH10  .                               4027
         SA,1     13,SRCH20  .                               4028
         SA,1     13,SRCH21  .                               4029
         SA,1     13,SRCH23  .                               4031
         SA,1     13,SRCH24  .                               4032
         LMJ      11,TACE$   . ACCEPT FRAME CT ERRORS        4036
SRCH3    +0                  . UNIT CODE                     4037
SRCH4    LMJ      11,TPS$    . PASS FM BACKWARD              4038
         +0                  . UNIT CODE                     4039
         J        SRCH5A     .                               40391
SRCH5    LMJ      11,TPE$    . PASS FM FORWARD               4040
         +0                  . UNIT CODE                     4041
SRCH5A   LMJ      11,TRF$    . READ FILE NBR                 4042
SRCH6    +0                  . UNIT                          4043
         +1,NBRBUF           . ACCESS WD                     4044
SRCH7    LMJ      11,TCHK$   . CHECK STAT                    4045
         +MERR$,0            . ERROR, UNIT                   4046
         +SRCH5A,SRCH7       . FM, IN PROCESS                4047
         SSL      13,18      . CLEAR LHW                     4048
         JNZ      13,SRCH31  . ACCESS WD NOT ZERO            4049
         LA,7     13,NBRBUF  . GET FILE NBR FROM TAPE        4050
         TNE      13,(-0)    . IS IT EOT                     4051
         J        SRCH9      . YES                           4052
         LX       11,SAV11   . RESTORE X11                   40521
         LA       14,*0,11   . GET FILE NBR- INPUT           4053
         TNE      14,13      . COMPARE FILE NBRS             4054
         J        3,11       . EQUAL, EXIT NORMALLY          4056
         TG       14,13      . IS TAPE HIGH                  4057
         J        SRCH5      . LESS, ADVANCE TAPE            4058
         LMJ      11,TRB$    . PASS FILE NBR                 4059
SRCH8    +0                  . UNIT                          4060
         +1,NBRBUF           . ACCESS WD                     40601
         LMJ      11,TRB$    . PASS FILE MARK                40602
SRCH8A   +0                  . UNIT                          40603
         +1,NBRBUF           . ACCESS WD                     40604
         LMJ      11,TCHK$   .                               40605
SRCH8B   +MERR$,0            .                               40606
         +$+1,$-2            .                               40607
         J        SRCH4      . TRY AGAIN                     4061
```

TABLE 5.10 - continued

LISTING OF SUBROUTINE SEARCH

```
SRCH9    LA        16,67 . REWOUND FLAG                    4062
         JNZ       16,SRCH11 . HAS IT BEEN REWOUND          40621
         LMJ       11,TRWS   . NO. REWIND                   4063
SRCH10   +0                  . UNIT                         4064
         LR,016    67,1      . REWOUND FLAG                 4065
         J         SRCH5     . TRY AGAIN                    4066
SRCH11   PSRINT  SRCH12,6    .                              4067
         J         SRCH13                                   4068
SRCH12   'CANNOT LOCATE REQUESTED FILE NBR'                 40681
SRCH13   LMJ       11,TRBS   . READ BACKWARDS 3 TIMES       4069
SRCH20   +0                  . PASS END SENTINAL            4070
         +1,NBRBUF           .                              4071
         LMJ       11,TRBS   . PASS FM                      4072
SRCH21   +0                  .                              4073
         +1,NBRBUF           .                              4074
         LMJ       11,TRBS   . READ FILE NBR                4075
SRCH23   +0                  . UNIT                         4076
         +1,NBRBUF           . ACCESS WD                    4077
         LMJ       11,TCHKS  . TAPE CHECK                   4078
SRCH24   +MERRS,0                                           4079
         +S+1,S-2 .
FF       FORM      6,12,18                                  4081
         LX        11,SAV11  .                              40811
         LA        12,*0,11  .                              4082
         LMJ       11,EBOS                                  4083
         FF        6,5,SRH25A+4                             4084
SRCH25   PSRINT  SRH25A,6    .                              4085
         J         SRH25B    .                              4086
SRH25A   'REQUESTED FILE NBR =            OCTAL'            40861
SRH25B   LA,6      12,NBRBUF . LAST NUMBER ON TAPE          4087
         ANA,016   12,1      . COMPUTE NEXT TO LAST         40871
         LMJ       11,EBOS   .                              4088
         FF        6,5,SRH26A+4                             4089
SRCH26   PSRINT  SRH26A,6    .                              4090
         J         SRCH35    .                              4091
SRH26A   'LAST FILE NBR ON TAPE =         OCTAL'            4092
SRCH31   PSRINT    SRCH32,7  .                              4107
         J         SRCH33    .                              4108
SRCH32   'FILE MARK NOT FOLLOWED BY FILE NUMBER'            4109
```

TABLE 5.10 - continued

LISTING OF SUBROUTINE SEARCH

```
SRCH33    LX        11,SAV11                                   4110
          LA        12,*0,11    . GET FILE NBR                 4111
          LMJ       11,EBO$     .                              4112
          FF        6,5,SRCH34+4 .                             41121
          P$RINT SRCH34,6       .                              41122
          J         SRCH35                                     41123
SRCH34    'FILE TO BE LOCATED WAS          OCTAL'              41124
SRCH35    LA        12,LUN      .                              41125
          SA,8      12,SRCH36+3 .                              41126
          P$RINT    SRCH36,4    .                              41127
          J         MEXIT$      .                              41128
SRCH36    'ERROR ON LOGICAL UNIT    '                          41129
$(1),FLNBR +1,1,1 . OUTPUT WRD                                 4113
LUN       +0 .                                                 41131
SAV11     +0 .                                                 4114
NBRBUF    +0 .                                                 4115
TBUF*     RES       175         . TAPE DUMP BUF                4116
          END                   . SRCH
```

TABLE 5.11

LISTING OF SUBROUTINE WFILE

```
         LA         12,*0,11    . GET FILE NBR              W05011
         SA,6       12,FLNBR    .                           W05012
         SA,7       12,FLNBR    .                           W05013
         LMJ        11,TEF$     . WRITE FM                  W05014
WF1      +0 .                                               W05015
         LMJ        11,TWR$     . WRITE FILE NBR            W05016
WF2      +0 .                                               W05017
         +1,FLNBR               .                           W05018
WF3      LMJ        11,TCHK$    .                           W05019
         +MERR$,0               .                           W05020
         +$+1,WF3               .                           W05021
         LMJ        11,TEF$     . WRITE FM                  W05022
WF4      +0 .                                               W05023
         LMJ        11,TWR$     . WRITE EOF                 W05024
WF5      +0 .                                               W05025
         +1,EOF                                             W05026
         LMJ        11,TRB$     . READ BACKWARD TEICE       W05027
WF6      +0 .                                               W05028
         +1,BUF1  .                                         W05029
         LMJ        11,TRB$ .                               W050291
WF6A     +0 .                                               W050292
         +1,BUF1                .                           W050293
         LMJ        11,TCHK$    .                           W050294
WF7      +MERR$,0                                           W05030
         +$+1,$-2               .                           W05031
WF8      LX         11,SAV11    .                           W05032
         J          3,11        . EXIT                      W05033
EOF + 0777777770000  .
SAV11    +0 .                                               W05034
FLNBR    +0 .                                               W05035
BUF1     +0 .                                               W050351
LUN      +0 .                                               W05037
         END . WFILE
```

TABLE 5.11 - continued

LISTING OF SUBROUTINE WFILE

```
/
WFILEF*   LA       12,*1,11 .  GET LUN
          LA       12,NTAB$,12
          SA       12,LUN .
          J        WFILE+2 .
          . ARGUMENTS ARE FILE NBR AND LUN IN THAT ORDER              W05001
          . BUT MUST BE FOLLOWED BY A NOP FOR A TOTAL OF 3 WORDS      W050011
WFILE*    LA       12,*1,11 . GET LUN                                 W05002
          SA       12,LUN                                             W050022
          SX       11,SAV11 .                                         W050023
          LA       12,LUN .                                           W05003
          SA,1     12,WF7                                             W05004
          SA,1     12,WF1     . STORE LUN                             W05005
          SA,1     12,WF2     .                                       W05006
          SA,1     12,WF3+1   .                                       W05007
          SA,1     12,WF4     .                                       W05008
          SA,1     12,WF5     .                                       W05009
          SA,1     12,WF6     .                                       W05010
          SA,1     12,WF6A .                                          W050101
```

6.0 DATA CARDS

The data cards required to run the Masonboro hydraulic model are given in Table 6.1. A discussion of the parameters follows.

6.1 Card 1

IMAX: is the number of columns in the cell structure, i.e., the number of increments along the x-axis.

JMAX: is the number of rows in cell structure, i.e., the number of increments along the y-axis.

NREEF: is the number of cell walls that are designated as barriers.

IECHO: this parameter indicates what data is to be echo printed, if any. The options available are

0 - no echo print of input data
1 - echo print input data and calculated friction factors
2 - echo print input data only.

ITAPE: tape file number to which values of IFLAG , H , QX , and QY , and other control information are written. If equal to zero (0), no output to tape is performed. The tape unit number is one (1).

IQYQXH: this parameter controls the initialization of the water height, the flow through the cell walls, and other parameters. If it is positive, a restart is assumed and the input if from tape and IQYQXH

TABLE 6.1

DATA CARDS

CARD 1: FORMAT(16I5)

IMAX,JMAX,NREEF,IECHO,ITAPE,IQYQXH,IPRT,IPRMOD,NSTEP,JB,
JBR,IBU,IOPP,IOPP1,IPCH

CARD 2: FORMAT(7F10.3)

DT,TMAX,PTIME,DS,TTAPE,VAL,TLIM

CARD 3: FORMAT(6F10.3)

DELTA,DELTA2,TIME,TINT,TPRMB,TPRME

BARRIER CARDS: FORMAT(3F10.0)

CS(NREEF) ZB(NREEF) DYB(NREEF)

CARDS FROM 2 TO NREEF-1

CS(1) ZB(1) DYB(1)

CORIOLIS CARD: FORMAT (2F10.0)

OMEGA PHI

SEABED ELEVATION CARDS: FORMAT(20F4.0)

Z(21,2)··

Z(1,2) Z(2,2) Z(3,2)·····················Z(19,2) Z(20,1)

Z(41,1) Z(42,1)··········Z(46,1)

Z(21,1) Z(22,1)··························Z(39,1) Z(40,1)

Z(1,1) Z(2,1) Z(3,1)·····················Z(19,1) Z(20,1)

TABLE 6.1 - Continued

FRICTION SCALING CARDS: FORMAT(11F7.3)

```
F(12,2)...................................................
F(1,2) F(2,2).....................................F(11,2)
F(23,1).........................F(30,1)
F(12,1)......................................F(22,1)
F(1,1) F(2,1).........................F(10,1) F(11,1)
```

FLAG FIELD CARDS: FORMAT(80I1)

```
CARD FOR LAST (#IMAX) COLUMN
CARD FOR COLUMNS 2 TO IMAX-1
CARD FOR FIRST COLUMN
CARD FOR LAST (#JMAX) ROW
CARD FOR ROWS 2 TO JMAX-1
CARD FOR FIRST ROW
```

INITIALIZATION CARDS: FORMAT(8E10.4) (IF USED)

```
QY(1,1)..........................................
QX(26,32) QX(27,32)..........................QX(46,32)
..............................................
QX(1,1) QX(2,1).................................QX(20,1)
H(26,32) H(27,32)....................H(45,32) H(46,32)
H(15,2) H(16,2)..............................H(34,2)
H(41,1).............H(46,1) H(1,2)..............H(14,2)
H(21,1)........................................H(40,1)
H(1,1) H(2,1)................................H(20,1)
```

TABLE 6.1 - Continued

WIND CARDS: FORMAT(I2,F10.1)

N ROT

FORMAT(3F10.2)

WSPEED(N) WDIR(N) WTIME(N)

CARDS FROM 3 TO N-1

WSPEED(2) WDIR(2) WTIME(2)

WSPEED(1) WDIR(1) WTIME(1)

TIDE CARDS: FORMAT(I10,F10.0)

NDAY DSHIFT

FORMAT(A4,A6,F6.0,4X,12F4.2/20X,12F4.2)

column 21

TD(12)........................TD(23)

LOC DATE TIMCOR TD(0)........................TD(11)

PRINT CARDS: FORMAT(I5)

IPL

FORMAT(10(A6,A1))

TITLE 11 TITLE 12....................................

TITLE 1 TITLE 2..................................TITLE 10

FORMAT(16I5)

IP(17) JP(17)

IP(9) JP(9) IP(10) JP(10)..................IP(16) JP(16)

IP(1) JP(1) IP(2) JP(2)......................IP(8) JP(8)

is the file number. If it is negative, the input is from cards. If zero (0), no input is assumed, and the arrays are set to zero. The tape unit number is one (1). For more information, see initialization cards.

IPRT: this parameter controls the form of the output at the time interval PTIME (see Card 2).

= 1 output through subroutine MODPRI

= 2 output through subroutine PRINT1

= 3 output through subroutines MODPRI and PRINT1

IPRMOD: number of calls to subroutine MOD between calls to subroutine MODPRI from subroutine MOD.

NSTEP: number of time steps between calls to subroutine MOD from program MASON.

JB: smallest J index of tidal flat cell.

JBR: smallest J index of cells with special boundary on right cell wall and that is not an external cell.

IBU: smallest I index of cells with special boundary on top cell wall and that is not an external cell.

IOPP: print control parameter sent to subroutine MODPRI by subroutine MOD. For values see section on subroutine MODPRI (Section 5.8).

IOPP1: print control parameter sent to subroutine MODPRI by call in main program controlled by parameter IPRT (see Section 5.8 and Table 5.8).

IPCH: a second print control parameter used in subroutine PRINT1 to control the form of the output (see Section 5.8).

6.2 Card 2

DT and DS: the time increment, DT , and cell size, DS , are dependent on each other. The dependence is governed by the numerical stability requirement given by equation (16) in Section 2.3. The equation is

$$DT \leq \frac{DS}{\sqrt{2g\ D_{max}}}$$

where g is the acceleration due to gravity and D_{max} is the maximum water depth expected. Using a one nautical mile grid, DS = 6080 feet, the time increment DT is at most approximately 2 minutes.

TMAX: is the length of the run in hours.

PTIME: this is the time interval at which H , QY , and QX are printed for up to seventeen (17) control cells (see print cards).

DS: see DT above.

TTAPE: time interval in minutes between dumps to tape starting with file ITAPE.

VAL: friction factor used in tidal flats.

TLIM: absolute time (not relative) in hours that run is to stop.

6.3 Card 3

DELTA: discrimination value in feed used to decide whether a cell is flooded or not (see Section 5.4).

DELTA2: similar to DELTA but used to determine if there is flow across a barrier.

TIME: absolute time at start of program. If restarted from tape this value is overridden by the value on the tape.

TINT: initial water level. If restarted from tape values on tape used.

TPRMB: time in hours after which tidal prism can be calculated.

TPRME: time in hours before which tidal prism can be calculated.

$$TPRMB < TPRME$$

These values should bracket the period that the tide is rising for which the tidal prism is desired.

6.4 Barrier Cards

The three parameters on each card are

- C_s - the barrier discharge coefficients (see page 9 in Section 2.2),
- ZB - the barrier elevations with respect to datum level in feet, and

DYB - the barrier widths in feet. Not used in this program.

These cards specify the parameters for the various barriers. There should be one card for each cell wall designated as a barrier. If there are two barriers for a given cell, the barrier perpendicular to the flow in the x direction (right wall barrier) is treated first and then the other. The cards are arranged in the order in which the barriers are encountered by the program while it is stepping through the grid structure.

6.5 Coriolis Card

OMEGA = 0.0000729 rad/sec

PHI = latitude in deg of site.

6.6 Seabed Elevation Cards

These specify the bottom elevations with respect to datum level. Elevations below the datum are input as positive numbers while those above the datum are negative numbers. Elevations for external cells should be 99.9. They are punched on the cards by row, see page 195 and 4) in Section 2.3, page 15.

6.7 Friction Scaling Cards

The values on these cards are multiplied times the base values of the friction factors to modify them. The base value for non-tidal flat cells is .0025 and that for tidal flat cells is VAL specified on the second data card. They are punched on the cards by row like the seabed elevation cards.

6.8 Initialization Cards

These cards specify the initial values of H , QX , and QY . They need to be present only if the values are not read from tape (IQYQXH > 0) or are not set equal to zero (IQYQXH = 0). If the initial values are read from cards, they are read in the following order: 1) the water level array, H ; 2) the flow per unit width in the x direction, QX ; and 3) the flow per unit width in the y direction, QY . Each array is read in its entirety before the next array is read.

The array elements of each array are read in the order in which the cells are processed by the program (see end of Section 2.3). For a given array, arrange the rows side by side to form a one-dimensional array. That is, place row two (2) to the right of row one (1), row three (3) to the right of row (2), and so on. Next, punch the values on the cards using the required format starting with the first value of row one (1) and continuing until the last value has been punched.

6.9 Flag Field Cards

These cards specify the flag field for the program. There is a one-to-one correspondence between the flags and the cell walls.

In specifying the flags, the flags for the cell walls perpendicular to the x-direction (vertical walls are specified first). Starting with cell (1,1) the condition of all the vertical cell walls in row one are specified as they are encountered going in a positive x-direction. Then the second row is done the same way and so on until the last row is done. Each row goes on a separate

card. After all of the rows are done then the horizontal cell walls are encoded in a similar manner using the columns. Starting with cell (1,1) again, the flags for the walls are specified as the walls are encountered going up the column. After the first column is done the second is done, and so on. Each column is contained on one card.

All cell walls are encoded including the outer walls. Therefore, if the grid is N by M , (N,M), there will be N+1 numbers for the row cards and M+1 numbers on the column cards.

The cell walls are encoded as if all flood plains were completely flooded, i.e., flag = 0. The flag system is discussed in Section 3.3 and the flag key is given in Table 3.2. The flag cards are omitted in restart runs.

6.10 Wind Cards

These cards specify the changes in the wind. A change in the wind is defined as a change in either the wind speed or wind direction. The wind changes are currently treated as step functions. No interpolation is performed for times between the wind changes. The program will currently accept up to ten (10) wind changes (see Section 5.6).

The variables are

N - number of wind changes. If set equal to zero, none of the remaining variables are required and calm conditions are assumed. The cards

normally containing WSPEED , WDIR , and WTIME should be omitted.

ROT - the angle between north and the positive x-axis. measured in a clockwise direction starting at north. For the GB model ROT = 51.5^{o} while for the P. H. Robinson and Cedar Bayou model ROT equals 100^{o} and 41^{o}, respectively.

WSPEED - the new wind speed (mph).

WDIR - the new wind direction with respect to north in a clockwise direction. This is the direction reported by the U. S. Weather Bureau.

WTIME - the time in minutes at which the changes occurred.

6.11 Tide Cards

The actual half-hourly record always starts in column 21 of the data cards with four cards per day of record. Each day has associated with it a LOC , DATE , DSHIFT , and a TIMCOR .

The variable definitions are

N - the number of days of input tidal records.

DSHIFT - datum correction for input record in feet. This is included so that if after the tide cards have been punched the reference datum is changed the tide cards need not be repunched. The error introduced by the change need only be entered here.

LOC - four (4) alphanumeric characters used to identify the input record

DATE - the calendar date of record in the form MMDDYY.

TIMECOR - time correction for date in minutes. Its use is similar to *DSHIFT above.*

TD - actual half-hourly record of tide level in feet with respect to mean sea level. Six (6) hours of data appear on one card. The first six (6) hours of a day are on the same card as LOC , DATE , and TIMECOR . The remainder are on the next three (3) cards.

6.12 Print Cards

These cards are used to identify the cells for which print out is desired (see Section 3.6). The variable definitions are

IPT - the number of cells for which information is to be printed. IPL must not exceed seventeen (17).

TITLE - a string of seven (7) alphanumeric characters used to identify the cells on the output for which the information is to be printed.

IP & JP - arrays of the cell coordinates for the location of the cells. The values for a given cell are adjacent to each other on the data card with the I subscript first.

6.13 Data Deck Arrangement

To run the program one only needs to assemble the program deck along with the proper control cards and data deck. The data card arrangement in the data deck is the same as in Table 6.1 at the beginning of this section. Tables 6.2 and 6.3 are listings of typical data decks used in the verification runs. The data deck listed in Table 6.2 is for an initial start with tape dumps to a clean tape. If the tape is not a clean tape, and it is desired to save the files on the tape, the only difference would be that the variable ITAPE (circled in listing) would be equal to the last file that it is desired to save plus one. This assumes that the tape was written using SEARCH and WFILE.

The data deck listed in Table 6.3 is for a restart from tape file number ten on a tape. Note that in the restart deck there are no flag cards present. When restarting, the input file should have approximately the same water levels during the same phase of the tidal cycle as the restart time.

TABLE 6.2

EXAMPLE OF DATA DECK FOR INITIAL VERIFICATION RUN

```
   30   25    1    5    1    0    3   10   36   13   16   15   16   29
.083333333        4.0       30.0       300.        60.      .0015      45.5
        .1         .1       26.5       .590       37.5       44.0
       1.0         .5
       0.0        0.0
32.032.032.032.030.025.020.012.011.010.010.014.018.023.023.028.028.028.028.028.0
34.034.034.034.034.034.034.034.034.034.0
32.032.032.030.027.024.018.012.011.0 9.5 8.510.018.021.021.023.023.023.028.028.0
32.034.034.034.034.034.034.034.034.034.0
29.029.028.026.023.018.013.011.0 9.5 8.0 5.5 5.610.614.314.016.018.520.023.028.0
32.032.032.032.032.032.032.032.032.032.0
28.026.023.022.017.016.011.010.0 8.0 8.0 7.5 4.2 7.218.010.013.015.019.023.024.0
27.028.028.028.029.028.028.029.029.029.0
24.022.020.017.015.012.011.010.0 8.0 8.0 6.5 4.0 3.410.719.310.014.018.020.020.5
21.022.024.025.025.025.025.027.028.028.0
20.019.016.015.013.011.010.0 9.0 8.0 6.0 6.0 3.0 1.0 6.116.7 8.511.514.017.017.0
19.020.020.021.021.522.023.525.027.027.0
17.015.513.513.012.010.0 9.0 8.5 8.0 7.0 5.5 3.0 1.0 2.415.814.4 8.011.014.015.0
15.517.018.019.520.521.022.024.025.025.0
15.014.013.012.511.510.0 9.0 8.5 8.0 8.0 6.0 4.0 1.0 1.010.616.4 6.0 7.0 9.013.0
13.515.016.517.519.020.521.523.024.023.0
13.513.011.511.510.510.0 9.0 8.0 7.0 8.0 5.0 4.0 1.5 1.5 8.018.4 7.5 3.0 7.511.0
13.014.015.016.017.018.020.021.021.521.0
12.011.010.510.0 9.5 9.0 8.0 7.0 7.0 7.0 5.0 3.0 1.0 3.1 9.817.6 9.6 2.0 2.0 5.0
 9.011.511.513.015.017.019.018.018.018.0
10.0 9.0 9.0 8.5 8.0 7.0 6.0 6.0 5.0 6.0 5.0 2.0 1.5 6.412.516.8 7.199.9 2.0 2.0
 3.0 4.0 5.0 6.0 7.0 8.0 8.0 8.0 9.010.0
 6.0 5.0 5.0 4.0 3.0 3.0 3.0 4.0 4.0 5.0 6.0 4.0 2.0 6.717.814.9 4.6 0.299.999.9
99.999.999.9 2.0 2.0 2.0 3.0 4.0 4.0 3.0
99.999.999.999.999.999.999.999.999.999.9-0.5 5.0 3.0 5.127.013.7 1.599.999.999.9
99.999.999.999.999.999.999.999.999.999.9
99.999.999.999.999.999.999.999.999.999.999.9-1.0 0.5 5.826.0 9.9-1.099.999.999.9
99.999.999.999.999.999.999.999.999.999.9
99.999.999.999.999.999.999.999.999.999.999.999.9-1.0 8.719.0 8.299.999.999.999.9
99.999.999.999.999.999.999.999.999.999.9
99.999.999.9-2.3-2.3-2.0-1.5-1.5-1.599.999.999.9 3.011.823.5 8.299.999.999.9 1.9
 1.7 1.4 3.5 5.6 6.4 5.6 6.3 8.0 8.0 8.0
```

TABLE 6.2 - continued

EXAMPLE OF DATA DECK FOR INITIAL VERIFICATION RUN

```
-1.8-1.8-1.7-1.3-1.0 0.8 2.0 2.9 3.1 2.7 4.3 5.6 5.0 5.420.714.5 9.7 9.510.711.4
11.611.812.313.215.817.317.617.013.513.0
 3.0 3.3 3.3 4.6 5.1 6.1 6.3 6.4 6.0 5.6 4.3 2.1-1.0 3.112.713.2 5.5 2.2 1.9 1.9
 3.8 8.712.411.910.8 6.0 6.0 8.0 1.5 2.0
 4.0 4.0 4.6 4.1 3.8 2.2 1.5-1.5 0.5 0.5-1.0-1.5-1.5 0.911.7 7.0 6.1 5.1 5.3 4.8
 3.6 2.7 1.2 0.599.999.9-0.5-0.5-0.5-0.6
-3.0-3.0-2.9-2.8-2.899.999.9-1.2-1.0-1.5-1.5-0.8-0.7-1.011.7 8.9-0.5 1.399.999.9
99.999.999.999.999.999.999.999.999.999.9
99.999.999.999.999.999.999.999.999.9-1.0-0.7-0.6-0.5-0.411.9 8.2-1.0-2.099.999.9
99.999.999.999.999.999.999.999.999.999.9
99.999.999.999.999.999.999.999.999.999.9-1.5-2.5-3.5-2.0 8.912.299.999.999.999.9
99.999.999.999.999.999.999.999.999.999.9
99.999.999.999.999.999.999.999.999.999.999.999.999.9-3.5 5.918.0-1.4-1.999.999.9
99.999.999.999.999.999.999.999.999.999.9
99.999.999.999.999.999.999.999.999.999.999.999.999.999.9 3.816.2-1.5-1.999.999.9
99.999.999.999.999.999.999.999.999.999.9
99.999.999.999.999.999.999.999.999.999.999.999.999.999.9 3.018.0-1.6-2.099.999.9
99.999.999.999.999.999.999.999.999.999.9
666666666666666666666666666666
666666640000000000002666666666
666666640000070000002666666666
666666640000070000002666666666
666666640000007000002666666666
666666640000000700002666666666
666666640000000700002666666666
666666640000000700002666666666
666666640000000070002666666666
666666640000000070002666666666
666666640000000077002666666666
666666640000000007666666666666
666666666670000007666666666666
666666666670000007666666666666
666666666666700076666666666666
666700000766700076667000000003
630000000000000000000000000003
630000000000000000000000000003
630000000000000000000017670003
630007670000000000766666666666
666666666700000000766666666666
```

TABLE 6.2 - continued

EXAMPLE OF DATA DECK FOR INITIAL VERIFICATION RUN

```
6666666666700000766666666666666
6666666666666700003666666666666
6666666666666670003666666666666
6666666666666670003666666666666
66666666666666666666666666
66666666666666667000766666
66666666666666667000766666
66666666666666670000766666
66666666666666670000766666
66666666666666670007666666
66666666666666670007666666
64000000000076670000766666
64000000000076670000766666
64000000000076667000076666
64000000000007667000007666
64000000000000767000007666
64000000000000000000007666
64000000000000000000000766
64007000000000000000000003
64000700000000000000000003
64000000700000767000077003
64000000007776667000077003
64000000000766667007666666
64000000000766667007666666
66000000000666670007666666
66666666666666670007666666
66666666666666670007666666
66666666666666670007666666
66666666666666670076666666
66666666666666670076666666
66666666666666670003666666
66666666666666670003666666
66666666666666670007666666
66666666666666670003666666
 0
```

TABLE 6.2 - continued

EXAMPLE OF DATA DECK FOR INITIAL VERIFICATION RUN

```
          2
MAS 110969      1    0.290.370.590.951.371.792.202.723.183.623.994.29
                2    4.424.434.324.103.783.402.902.381.881.390.950.57
                3    0.350.290.310.460.901.271.702.302.843.303.643.90
                4    4.224.414.414.203.843.543.052.632.191.751.340.94
MAS 120969      1    0.550.340.290.370.590.951.371.792.202.733.183.62
                2    4.004.304.424.434.334.103.793.402.892.381.881.40
                3    0.950.570.350.290.310.460.901.271.702.302.843.30
                4    3.643.904.224.414.414.203.843.543.052.632.291.75
   11
(15,7) (16,7) (18,17)(19,17)(3,18) (3,19) (15,23)(16,23)(28,16)(28,17.
(28,18)
   15    7   16    7   18   17   19   17    3   18    3   19   15   23   16   23
   28   16   28   17   28   18
```

TABLE 6.3

EXAMPLE OF DATA DECK FOR RESTART OF VERIFICATION RUN

```
  30   25    1    5   10    9    3   10   36   13   16   15   16   29
.083333333        4.0       30.0      300.      60.      .0015     45.5
        .1         .1       26.5      .590      37.5      44.0
       1.0        .5
       0.0       0.0
32.032.032.032.030.025.020.012.011.010.010.014.018.023.023.028.028.028.028.028.0
34.034.034.034.034.034.034.034.034.034.0
32.032.032.030.027.024.018.012.011.0 9.5 8.510.018.021.021.023.023.023.028.028.0
32.034.034.034.034.034.034.034.034.034.0
29.029.028.026.023.018.013.011.0 9.5 8.0 5.5 5.610.614.314.016.018.520.023.028.0
32.032.032.032.032.032.032.032.032.032.0
28.026.023.022.017.016.011.010.0 8.0 8.0 7.5 4.2 7.218.010.013.015.019.023.024.0
27.028.028.028.029.028.028.029.029.029.0
24.022.020.017.015.012.011.010.0 8.0 8.0 6.5 4.0 3.410.719.310.014.018.020.020.5
21.022.024.025.025.025.025.027.028.028.0
20.019.016.015.013.011.010.0 9.0 8.0 6.0 6.0 3.0 1.0 6.116.7 8.511.514.017.017.0
19.020.020.021.021.522.023.525.027.027.0
17.015.513.513.012.010.0 9.0 8.5 8.0 7.0 5.5 3.0 1.0 2.415.814.4 8.011.014.015.0
15.517.018.019.520.521.022.024.025.025.0
15.014.013.012.511.510.0 9.0 8.5 8.0 8.0 6.0 4.0 1.0 1.010.616.4 6.0 7.0 9.013.0
13.515.016.517.519.020.521.523.024.023.0
13.513.011.511.510.510.0 9.0 8.0 7.0 8.0 5.0 4.0 1.5 1.5 8.018.4 7.5 3.0 7.511.0
13.014.015.016.017.018.020.021.021.521.0
12.011.010.510.0 9.5 9.0 8.0 7.0 7.0 7.0 5.0 3.0 1.0 3.1 9.817.6 9.6 2.0 2.0 5.0
 9.011.511.513.015.017.019.018.018.018.0
10.0 9.0 9.0 8.5 8.0 7.0 6.0 6.0 5.0 6.0 5.0 2.0 1.5 6.412.516.8 7.199.9 2.0 2.0
 3.0 4.0 5.0 6.0 7.0 8.0 8.0 8.0 9.010.0
 6.0 5.0 5.0 4.0 3.0 3.0 3.0 4.0 4.0 5.0 6.0 4.0 2.0 6.717.814.9 4.6 0.299.999.9
99.999.999.9 2.0 2.0 2.0 3.0 4.0 4.0 3.0
99.999.999.999.999.999.999.999.999.999.9-0.5 5.0 3.0 5.127.013.7 1.599.999.999.9
99.999.999.999.999.999.999.999.999.999.9
99.999.999.999.999.999.999.999.999.999.999.9-1.0 0.5 5.826.0 9.9-1.099.999.999.9
99.999.999.999.999.999.999.999.999.999.9
99.999.999.999.999.999.999.999.999.999.999.999.9-1.0 8.719.0 8.299.999.999.999.9
99.999.999.999.999.999.999.999.999.999.9
99.999.999.9-2.3-2.3-2.0-1.5-1.5-1.599.999.999.9 3.011.823.5 8.299.999.999.9 1.9
 1.7 1.4 3.5 5.6 6.4 5.6 6.3 8.0 8.0 8.0
```

TABLE 6.3 - continued

EXAMPLE OF DATA DECK FOR RESTART OF VERIFICATION RUN

```
-1.8-1.8-1.7-1.3-1.0 0.8 2.0 2.9 3.1 2.7 4.3 5.6 5.0 5.420.714.5 9.7 9.510.711.4
11.611.812.313.215.817.317.617.013.513.0
 3.0 3.3 3.3 4.6 5.1 6.1 6.3 6.4 6.0 5.6 4.3 2.1-1.0 3.112.713.2 5.5 2.2 1.9 1.9
 3.8 8.712.411.910.8 6.0 6.0 8.0 1.5 2.0
 4.0 4.0 4.6 4.1 3.8 2.2 1.5-1.5 0.5 0.5-1.0-1.5-1.5 0.911.7 7.0 6.1 5.1 5.3 4.8
 3.6 2.7 1.2 0.599.999.9-0.5-0.5-0.5-0.6
-3.0-3.0-2.9-2.8-2.899.999.9-1.2-1.0-1.5-1.5-0.8-0.7-1.011.7 8.9-0.5 1.399.999.9
99.999.999.999.999.999.999.999.999.999.9
99.999.999.999.999.999.999.999.999.9-1.0-0.7-0.6-0.5-0.411.9 8.2-1.0-2.099.999.9
99.999.999.999.999.999.999.999.999.999.9
99.999.999.999.999.999.999.999.999.999.9-1.5-2.5-3.5-2.0 8.912.299.999.999.999.9
99.999.999.999.999.999.999.999.999.999.9
99.999.999.999.999.999.999.999.999.999.999.999.999.9-3.5 5.918.0-1.4-1.999.999.9
99.999.999.999.999.999.999.999.999.999.9
99.999.999.999.999.999.999.999.999.999.999.999.999.999.9 3.816.2-1.5-1.999.999.9
99.999.999.999.999.999.999.999.999.999.9
99.999.999.999.999.999.999.999.999.999.999.999.999.999.9 3.018.0-1.6-2.099.999.9
99.999.999.999.999.999.999.999.999.999.9
 0

           2
MAS 110969      1   0.290.370.590.951.371.792.202.723.183.623.994.29
                2   4.424.434.324.103.783.402.902.381.881.390.950.57
                3   0.350.290.310.460.901.271.702.302.843.303.643.90
                4   4.224.414.414.203.843.543.052.632.191.751.340.94
MAS 120969      1   0.550.340.290.370.590.951.371.792.202.733.183.62
                2   4.004.304.424.434.334.103.793.402.892.381.881.40
                3   0.950.570.350.290.310.460.901.271.702.302.843.30
                4   3.643.904.224.414.414.203.843.543.052.632.291.75
  11
(15,7) (16,7) (18,17)(19,17)(3,18) (3,19) (15,23)(16,23)(28,16)(28,17.
(28,18)
  15    7   16    7   18   17   19   17    3   18    3   19   15   23   16   23
  28   16   28   17   28   18
```

REFERENCES

Defant, Albert, Physical Oceanography, II, Pergamon Press, 1961.

Dronkers, J. J., Tidal Computations in Rivers and Coastal Waters, North Holland Publishing Company, 1964.

Espey, W. H., Jr., A. J. Hays, Jr., W. D. Bergman, J. P. Buckner, R. J. Huston, and G. H. Ward, Jr., "Galveston Bay Project Water Quality Modeling and Data Management Phase II Technical Progress Report," TRACOR Document Number T70-AU-7636-U, July, 1971, submitted to Texas Water Quality Board, Austin, Texas.

Harleman, D. R. F. and C. H. Lee, "The Computation of Tides and Currents in Estuaries and Canals," Tech. Bull. No. 16, Comm. on Tidal Hydraulics, U. S. Army Corps of Engineers, 1969.

Ippen, A. T., Ed., Estuary and Coastline Hydrodynamics, 1st ed., McGraw-Hill, 1966, pp 495, 623-625.

Kreiss, H. and J. Oliger, "Methods for the Approximate Solution of Time Dependent Problems," GARP Publications Series No. 10, WMO, Geneva, 1973.

Lazanoff, S. M., "An Evaluation of a Numerical Water Elevation and Tidal Current Prediction Model Applied to Monterey Bay," U. S. Naval Postgraduate School, 1971.

Leendertse, J. J., "A Water-Qaulity Simulation Model for Well-Mixed Estuaries and Coastal Seas: Volume 1, Principles of Computation," Memorandum RM-6230-RC, Feb., 1970, Rand Corporation, Santa Monica, California.

Leendertse, J. J., "Aspects of a Computational Model for Long-Period Water-Wave Propogation," Memorandum RM-5294-PR, May, 1967, Rand Corporation, Santa Monica, California.

Masch, F. D. et al., "A Numerical Model for the Simulation of Tidal Hydrodynamics in Shallow Irregular Estuaries," Tech. Report HYD 12-6901, Department of Civil Engineering, The University of Texas at Austin, 1969.

Neumann, G. and W. J. Pierson, Principles of Physical Oceanography, Prentice-Hall, Englewood Cliffs, New Jersey, 1966.

Platzman, G. W., "A Numerical Computation of the Surge of 26 June 1954 on Lake Michigan," Geophysica, Vol. 6, 1958, pp 407-438.

Reid, R. O. and B. R. Bodine, "Numerical Model for Storm Surges in Galveston Bay," Proceedings American Society of Civil Engineers, Vol. WW1, Feb., 1968, pp 33-57

Sverdrup, H. V., M. W. Johnson and R. H. Fleming, The Oceans, Prentice-Hall, Englewood Cliffs, New Jersey, 1942.

Wu, Jin, "Wind Stress and Surface Roughness of Air-Sea Interface," Journal of Geophysical Research, Vol. 74, No. 2, 15 January 1969, pp 444-455.

APPENDIX A

FLAG LISTING

The following is a complete listing of all possible flag combinations. Those lines marked with an '*' on the right either are not legal combinations or program logic will not handle them. The numbers given for bit position 6 to 4 are the octal equivalent for those three bits. Bits are numbered from the right.

The numbers in the right two columns are the decimal equivalent and are printed when the flag field is listed.

	CONDITION OF WALLS				DECIMAL EQUIVALENT	
	BOTTOM	LEFT	TOP	RIGHT	SOMETIMES	ALWAYS
BITS	8	7	6-4	3-1	FLOODED	FLOODED
	0	0	0	0	0	256
	0	0	0	1	1	257
	0	0	0	2	2	258
	0	0	0	3	3	259
	0	0	0	4	4	260*
	0	0	0	5	5	261
	0	0	0	6	6	262*
	0	0	0	7	7	263
	0	0	1	0	8	264
	0	0	1	1	9	265
	0	0	1	2	10	266
	0	0	1	3	11	267
	0	0	1	4	12	268*
	0	0	1	5	13	269
	0	0	1	6	14	270*
	0	0	1	7	15	271
	0	0	2	0	16	272
	0	0	2	1	17	273
	0	0	2	2	18	274
	0	0	2	3	19	275
	0	0	2	4	20	276*
	0	0	2	5	21	277
	0	0	2	6	22	278*
	0	0	2	7	23	279
	0	0	3	0	24	280
	0	0	3	1	25	281
	0	0	3	2	26	282
	0	0	3	3	27	283
	0	0	3	4	28	284*
	0	0	3	5	29	285
	0	0	3	6	30	286*
	0	0	3	7	31	287
	0	0	4	0	32	288*
	0	0	4	1	33	289*
	0	0	4	2	34	290*
	0	0	4	3	35	291*
	0	0	4	4	36	292*
	0	0	4	5	37	293*
	0	0	4	6	38	294*
	0	0	4	7	39	295*
	0	0	5	0	40	296
	0	0	5	1	41	297
	0	0	5	2	42	298
	0	0	5	3	43	299
	0	0	5	4	44	300*

	CONDITION OF WALLS				DECIMAL EQUIVALENT	
	BOTTOM	LEFT	TOP	RIGHT	SOMETIMES	ALWAYS
BITS	8	7	6-4	3-1	FLOODED	FLOODED
	0	0	5	5	45	301
	0	0	5	6	46	302*
	0	0	5	7	47	303
	0	0	6	0	48	304*
	0	0	6	1	49	305*
	0	0	6	2	50	306*
	0	0	6	3	51	307*
	0	0	6	4	52	308*
	0	0	6	5	53	309*
	0	0	6	6	54	310*
	0	0	6	7	55	311*
	0	0	7	0	56	312
	0	0	7	1	57	313
	0	0	7	2	58	314
	0	0	7	3	59	315
	0	0	7	4	60	316*
	0	0	7	5	61	317
	0	0	7	6	62	318*
	0	0	7	7	63	319
	0	1	0	0	64	320
	0	1	0	1	65	321
	0	1	0	2	66	322
	0	1	0	3	67	323
	0	1	0	4	68	324*
	0	1	0	5	69	325
	0	1	0	6	70	326*
	0	1	0	7	71	327
	0	1	1	0	72	328
	0	1	1	1	73	329
	0	1	1	2	74	330
	0	1	1	3	75	331
	0	1	1	4	76	332*
	0	1	1	5	77	333
	0	1	1	6	78	334*
	0	1	1	7	79	335
	0	1	2	0	80	336
	0	1	2	1	81	337
	0	1	2	2	82	338
	0	1	2	3	83	339
	0	1	2	4	84	340*
	0	1	2	5	85	341
	0	1	2	6	86	342*
	0	1	2	7	87	343

	CONDITION OF WALLS				DECIMAL EQUIVALENT	
	BOTTOM	LEFT	TOP	RIGHT	SOMETIMES FLOODED	ALWAYS FLOODED
BITS	8	7	6-4	3-1		
	0	1	3	0	88	344
	0	1	3	1	89	345
	0	1	3	2	90	346
	0	1	3	3	91	347*
	0	1	3	4	92	348
	0	1	3	5	93	349
	0	1	3	6	94	350*
	0	1	3	7	95	351
	0	1	4	0	96	352*
	0	1	4	1	97	353*
	0	1	4	2	98	354*
	0	1	4	3	99	355*
	0	1	4	4	100	356*
	0	1	4	5	101	357*
	0	1	4	6	102	358*
	0	1	4	7	103	359*
	0	1	5	0	104	360
	0	1	5	1	105	361
	0	1	5	2	106	362
	0	1	5	3	107	363*+
	0	1	5	4	108	364*
	0	1	5	5	109	365
	0	1	5	6	110	366*
	0	1	5	7	111	367
	0	1	6	0	112	368*
	0	1	6	1	113	369*
	0	1	6	2	114	370*
	0	1	6	3	115	371*
	0	1	6	4	116	372*
	0	1	6	5	117	373*
	0	1	6	6	118	374*
	0	1	6	7	119	375*
	0	1	7	0	120	376
	0	1	7	1	121	377
	0	1	7	2	122	378
	0	1	7	3	123	379*
	0	1	7	4	124	380*
	0	1	7	5	125	381
	0	1	7	6	126	382*
	0	1	7	7	127	383
	1	0	0	0	128	384
	1	0	0	1	129	385
	1	0	0	2	130	386
	1	0	0	3	131	387
	1	0	0	4	132	388*

	CONDITION OF WALLS				DECIMAL EQUIVALENT	
	BOTTOM	LEFT	TOP	RIGHT	SOMETIMES	ALWAYS
BITS	8	7	6-4	3-1	FLOODED	FLOODED
	1	0	0	5	133	389
	1	0	0	6	134	390*
	1	0	0	7	135	391
	1	0	1	0	136	392
	1	0	1	1	137	393
	1	0	1	2	138	394
	1	0	1	3	139	395
	1	0	1	4	140	396*
	1	0	1	5	141	397
	1	0	1	6	142	398*
	1	0	1	7	143	399
	1	0	2	0	144	400
	1	0	2	1	145	401
	1	0	2	2	146	402
	1	0	2	3	147	403
	1	0	2	4	148	404*
	1	0	2	5	149	405
	1	0	2	6	150	406*
	1	0	2	7	151	407
	1	0	3	0	152	408*
	1	0	3	1	153	409*
	1	0	3	2	154	410*
	1	0	3	3	155	411*
	1	0	3	4	156	412*
	1	0	3	5	157	413*
	1	0	3	6	158	414*
	1	0	3	7	159	415*
	1	0	4	0	160	416*
	1	0	4	1	161	417*
	1	0	4	2	162	418*
	1	0	4	3	163	419*
	1	0	4	4	164	420*
	1	0	4	5	165	421*
	1	0	4	6	166	422*
	1	0	4	7	167	423*
	1	0	5	0	168	424
	1	0	5	1	169	425
	1	0	5	2	170	426
	1	0	5	3	171	427
	1	0	5	4	172	428*
	1	0	5	5	173	429
	1	0	5	6	174	430*
	1	0	5	7	175	431
	1	0	6	0	176	432*
	1	0	6	1	177	433*

	CONDITION OF WALLS				DECIMAL EQUIVALENT	
	BOTTOM	LEFT	TOP	RIGHT	SOMETIMES	ALWAYS
BITS	8	7	6-4	3-1	FLOODED	FLOODED
	1	0	6	2	178	434*
	1	0	6	3	179	435*
	1	0	6	4	180	436*
	1	0	6	5	181	437*
	1	0	6	6	182	438*
	1	0	6	7	183	439*
	1	0	7	0	184	440
	1	0	7	1	185	441
	1	0	7	2	186	442
	1	0	7	3	187	443
	1	0	7	4	188	444*
	1	0	7	5	189	445
	1	0	7	6	190	446*
	1	0	7	7	191	447
	1	1	0	0	192	448
	1	1	0	1	193	449
	1	1	0	2	194	450
	1	1	0	3	195	451*
	1	1	0	4	196	452*
	1	1	0	5	197	453
	1	1	0	6	198	454*
	1	1	0	7	199	455
	1	1	1	0	200	456
	1	1	1	1	201	457
	1	1	1	2	202	458
	1	1	1	3	203	459*
	1	1	1	4	204	460*
	1	1	1	5	205	461
	1	1	1	6	206	462*
	1	1	1	7	207	463
	1	1	2	0	208	464
	1	1	2	1	209	465
	1	1	2	2	210	466*
	1	1	2	3	211	467*
	1	1	2	4	212	468*
	1	1	2	5	213	469*
	1	1	2	6	214	470*
	1	1	2	7	215	471*
	1	1	3	0	216	472*
	1	1	3	1	217	473*
	1	1	3	2	218	474*
	1	1	3	3	219	475*
	1	1	3	4	220	476*
	1	1	3	5	221	477*
	1	1	3	6	222	478*

	CONDITION OF WALLS				DECIMAL EQUIVALENT	
	BOTTOM	LEFT	TOP	RIGHT	SOMETIMES	ALWAYS
BITS	8	7	6-4	3-1	FLOODED	FLOODED
	1	1	3	7	223	479*
	1	1	4	0	224	480*
	1	1	4	1	225	481*
	1	1	4	2	226	482*
	1	1	4	3	227	483*
	1	1	4	4	228	484*
	1	1	4	5	229	485*
	1	1	4	6	230	486
	1	1	4	7	231	487
	1	1	5	0	232	488
	1	1	5	1	233	489
	1	1	5	2	234	490
	1	1	5	3	235	491*
	1	1	5	4	236	492*
	1	1	5	5	237	493
	1	1	5	6	238	494*
	1	1	5	7	239	495
	1	1	6	0	240	496
	1	1	6	1	241	497*
	1	1	6	2	242	498*
	1	1	6	3	243	499*
	1	1	6	4	244	500
	1	1	6	5	245	501*
	1	1	6	6	246	502
	1	1	6	7	247	503
	1	1	7	0	248	504
	1	1	7	1	249	505
	1	1	7	2	250	506*
	1	1	7	3	251	507*
	1	1	7	4	252	508
	1	1	7	5	253	509
	1	1	7	6	254	510
	1	1	7	7	255	511

In accordance with letter from DAEN-RDC, DAEN-ASI dated 22 July 1977, Subject: Facsimile Catalog Cards for Laboratory Technical Publications, a facsimile catalog card in Library of Congress MARC format is reproduced below.

Chen, R J

Comparison of numerical and physical hydraulic models, Masonboro Inlet, North Carolina; Appendix 3: Numerical simulation of hydrodynamics (Tracor) / by R. J. Chen and L. A. Hembree, Jr. Vicksburg, Miss. : U. S. Waterways Experiment Station ; Springfield, Va. : available from National Technical Information Service, 1977.

xiv, 254, 7 p. : ill. ; 27 cm. (GITI report - U. S. Army. Corps of Engineers ; 6, Appendix 3)

Prepared for U. S. Army Coastal Engineering Research Center under Contract DACW72-72-C-0029, by Tracor, Austin, Texas.

General investigation of tidal inlets; a program of research conducted jointly by U. S. Army Coastal Engineering Research Center, Fort Belvoir, Virginia, [and] U. S. Army Engineer Waterways Experiment Station, Vicksburg, Mississippi.

References: p. 253-254.

1. Hydraulic models. 2. Hydromechanics. 3. Masonboro Inlet, N. C. 4. Mathematical models. 5. Numerical simulation.

(Continued on next card)

Chen, R J

Comparison of numerical and physical hydraulic models, Masonboro Inlet, North Carolina; Appendix 3: Numerical simulation of hydrodynamics (Tracor) ... 1977. (Card 2)

6. Tidal flats. I. Hembree, L. A., joint author. II. Tracor, Inc. III. United States. Coastal Engineering Research Center. IV. United States. Waterways Experiment Station, Vicksburg, Miss. V. Series: United States. Army. Corps of Engineers. GITI report ; 6, Appendix 3.

GB454.T5.U5 no.6 Appendix 3